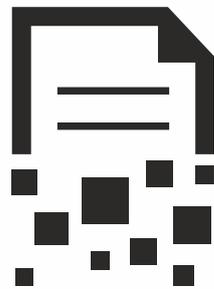


 **ACL 2013**

4 – 9 August | Sofia, Bulgaria

**51<sup>st</sup>**  
**ANNUAL MEETING OF THE  
ASSOCIATION FOR  
COMPUTATIONAL LINGUISTICS**



---

**Proceedings of  
the Conference**

---

**VOLUME 1: Long Papers**

ACL 2013

**51st Annual Meeting of the  
Association for Computational Linguistics**

**Proceedings of the Conference  
Volume 1: Long Papers**

August 4-9, 2013  
Sofia, Bulgaria

Production and Manufacturing by  
*Omnipress, Inc.*  
2600 Anderson Street  
Madison, WI 53704 USA

PLATINUM LEVEL SPONSOR



GOLD LEVEL SPONSORS



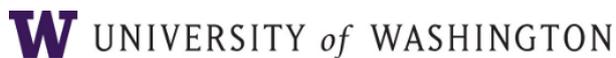
SILVER LEVEL SPONSORS



BRONZE LEVEL SPONSORS



SUPPORTER



BEST STUDENT PAPER AWARD



STUDENT VOLUNTEER



CONFERENCE BAG SPONSOR



CONFERENCE DINNER ENTERTAINMENT SPONSOR



LOCAL ORGANIZER



©2013 The Association for Computational Linguistics

Order copies of this and other ACL proceedings from:

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

ISBN 978-1-937284-50-3 (Volume 1)

## Preface: General Chair

Welcome to the 51st Annual Meeting of the Association for Computational Linguistics in Sofia, Bulgaria! The first ACL meeting was held in Denver in 1963 under the name AMTCL. This makes ACL one of the longest running conferences in computer science. This year we received a record total number of 1286 submissions, which is a testament to the continued and growing importance of computational linguistics and natural language processing.

The success of an ACL conference is made possible by the dedication and hard work of many people. I thank all of them for volunteering their time and energy in service to our community.

Priscilla Rasmussen, the ACL Business Manager, and Graeme Hirst, the treasurer, did most of the groundwork in selecting Sofia as the conference site, went through several iterations of planning and shouldered a significant part of the organizational work for the conference. It was my first exposure to the logistics of organizing a large event and I was surprised at how much expertise and experience is necessary to make ACL a successful meeting.

Thanks to Svetla Koeva and her team for their work on local arrangements, including social activities (Radka Vlahova, Tsvetana Dimitrova, Svetlozara Lesseva), local sponsorship (Stoyan Mihov, Rositsa Dekova), conference handbook (Nikolay Genov, Hristina Kukova), web site (Tinko Tinchev, Emil Stoyanov, Georgi Iliev), local exhibits (Maria Todorova, Ekaterina Tarpomanova), internet, wifi and equipment (Martin Yalamov, Angel Genov, Borislav Rizov) and student volunteer management (Kalina Boncheva). Perhaps most importantly, Svetla was the liaison to the professional conference organizer AIM Group, a relationship that is crucial for the success of the conference. Doing the local arrangements is a fulltime job for an extended period of time. We are lucky that we have people in our community who are willing to provide this service without compensation.

The program co-chairs Pascale Fung and Massimo Poesio selected a strong set of papers for the main conference and invited three great keynote speakers, Harald Baayen, Chantal Prat and Lars Rasmussen. Putting together the program of the top conference in our field is a difficult job and I thank Pascale and Massimo for taking on this important responsibility.

Thanks are also due to the other key members of the ACL organizing committees: Aoife Cahill and Qun Liu (workshop co-chairs); Johan Bos and Keith Hall (tutorial co-chairs); Miriam Butt and Sarmad Hussain (demo co-chairs); Steven Bethard, Preslav Nakov and Feiyu Xu (faculty advisors to the student research workshop); Anik Dey, Eva Vecchi, Sebastian Krause and Ivelina Nikolova (co-chairs of the student research workshop); Leo Wanner (mentoring chair); and Anisava Miltenova, Ivan Derzhanski and Anna Korhonen (publicity co-chairs).

I am particularly indebted to Roberto Navigli, Jing-Shin Chang and Stefano Faralli for producing the proceedings of the conference, a bigger job than usual because of the large number of submissions and the resulting large number of acceptances.

The ACL conference and the ACL organization benefit greatly from the financial support of our sponsors. We thank the platinum level sponsor, Baidu; the three gold level sponsors; the three silver level sponsors; and six bronze level sponsors. Three other sponsors took advantage of more creative options to assist us: Facebook sponsored the Student Volunteers; IBM sponsored the Best Student Paper Award; and SDL sponsored the conference bags. We are grateful for the financial support from these organizations.

Finally, I would like to express my appreciation to the area chairs, workshop organizers, tutorial presenters and reviewers for their participation and contribution.

Of course, the ACL conference is primarily held for the people who attend the conference, including the

authors. I would like to thank all of you for your participation and wish you a productive and enjoyable meeting in Sofia!

ACL 2013 General Chair  
Hinrich Schuetze, University of Munich

## Preface: Programme Committee Co-Chairs

Welcome to the 2013 Conference of the Association for Computational Linguistics! Our community continues to grow, and this year's conference has set a new record for paper submissions. We received 1286 submissions, which is 12% more than the previous record; we are particularly pleased to see a striking increase in the number of short papers submitted - 624, which is 21.8% higher than the previous record set in 2011.

Another encouraging trend in recent years is the increasing number of aspects of language processing, and forms of language, of interest to our community. In order to reflect this greater diversity, this year's conference has a much larger number of tracks than previous conferences, 26. Consequently, many more area chairs and reviewers were recruited than in the past, thus involving an even greater subset of the community in the selection of the program. We feel this, too, is a very positive development. We thank the area chairs and reviewers for their hard work.

A key innovation introduced this year is the presentation at the conference of sixteen papers accepted by the new ACL journal, Transactions of the Association for Computational Linguistics (TACL). We have otherwise maintained most of the innovations introduced in recent years, including accepting papers accompanied by supplemental materials such as corpora or software.

Another new practice this year is the presence of an industrial keynote speaker in addition to the two traditional keynote speakers. We are delighted to have as invited speakers two scholars as distinguished as Prof. Harald Baayen of Tuebingen and Alberta and Prof. Chantel Prat from the University of Wisconsin. Prof. Baayen will talk about using eye-tracking to study the semantics of compounds, an issue of great interest for work on distributional semantics. Prof. Prat will talk about research studying language in bilinguals using methods from neuroscience. The industrial keynote speaker, Dr. Lars Rasmussen from Facebook, will talk about the new graph search algorithm recently announced by the company. Last, but not least, the recipient of this year's ACL Lifetime Achievement Award will give a plenary lecture during the final day of the conference.

The list of people to thank for their contribution to this year's program is very long. First of all we wish to thank the authors who submitted top quality work to the conference; we would not have such a strong program without them, nor without the hard work of area chairs and reviewers, who enabled us to make often very difficult choices and to provide valuable feedback to the authors. As usual, Rich Gerber and the START team gave us crucial help with an amazing speed. The general conference chair Hinrich Schuetze provided valuable guidance and kept the timetable ticking along. We thank the local arrangements committee headed by Svetla Koeva, who played a key role in finalizing the program. We also thank the publication chairs, Jing-Shin Chang and Roberto Navigli, and their collaborator Stefano Faralli, who together produced this volume; and Priscilla Rasmussen, Drago Radev and Graeme Hirst, who provided enormously useful guidance and support. Finally, we wish to thank previous program chairs, and in particular John Carroll, Stephen Clark, and Jian Su, for their insight on the process.

We hope you will be as pleased as we are with the result and that you'll enjoy the conference in Sofia this Summer.

ACL 2013 Program Co-Chairs

Pascale Fung, Hong-Kong University of Science and Technology

Massimo Poesio, University of Essex



# Organizing Committee

## **General Chair:**

Hinrich Schuetze, University of Munich

## **Program Co-Chairs:**

Pascale Fung, The Hong Kong University of Science and Technology  
Massimo Poesio, University of Essex

## **Local Chair:**

Svetla Koeva, Bulgarian Academy of Sciences

## **Workshop Co-Chairs:**

Aoife Cahill, Educational Testing Service  
Qun Liu, Dublin City University & Chinese Academy of Sciences

## **Tutorial Co-Chairs:**

Johan Bos, University of Groningen  
Keith Hall, Google

## **Demo Co-Chairs:**

Miriam Butt, University of Konstanz  
Sarmad Hussain, Al-Khawarizmi Institute of Computer Science

## **Publication Chairs:**

Roberto Navigli, Sapienza University of Rome (Chair)  
Jing-Shin Chang, National Chi Nan University (Co-Chair)  
Stefano Faralli, Sapienza University of Rome

## **Faculty Advisors (Student Research Workshop):**

Steven Bethard, University of Colorado Boulder & KU Leuven  
Preslav I. Nakov, Qatar Computing Research Institute  
Feiyu Xu, DFKI, German Research Center for Artificial Intelligence

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

Anik Dey, The Hong Kong University of Science & Technology  
Eva Vecchi, Università di Trento

Sebastian Krause, DFKI, German Research Center for Artificial Intelligence  
Ivelina Nikolova, Bulgarian Academy of Sciences

**Mentoring Chair:**

Leo Wanner, Universitat Pompeu Fabra

**Publicity Co-Chairs:**

Anisava Miltenova, Bulgarian Academy of Sciences  
Ivan Derzhanski, Bulgarian Academy of Sciences  
Anna Korhonen, University of Cambridge

**Business Manager:**

Priscilla Rasmussen, ACL

**Area Chairs:**

Frank Keller, University of Edinburgh  
Roger Levy, UC San Diego  
Amanda Stent, AT&T  
David Suendermann, DHBW, Stuttgart, Germany  
Andrew Kehler, UC San Diego  
Becky Passonneau, Columbia  
Hang Li, Huawei Technologies  
Nancy Ide, Vassar  
Piek Vossen, Freie Universitat Amsterdam  
Philipp Cimiano, University of Bielefeld  
Sabine Schulte im Walde, University of Stuttgart  
Dekang Lin, Google  
Chiori Hori, NICT, Japan  
Keh-Yih Su, Behavior Design Corporation  
Roland Kuhn, NRC  
Dekai Wu, HKUST  
Benjamin Snyder, University of Wisconsin-Madison  
Thamar Solorio, University of Texas-Dallas  
Ehud Reiter, University of Aberdeen  
Massimiliano Ciaramita, Google  
Ken Church, IBM  
Carlo Strapparava, FBK  
Tomaz Erjavec, Jožef Stefan Institute  
Adam Przepiorkowski, Polish Academy of Sciences  
Patrick Pantel, Microsoft  
Owen Rambow, Columbia  
Chris Dyer, CMU  
Jason Eisner, Johns Hopkins  
Jennifer Chu-Carroll, IBM  
Bernardo Magnini, FBK  
Lluis Marquez, Universitat Politecnica de Catalunya

Alessandro Moschitti, University of Trento  
Claire Cardie, Cornell  
Rada Mihalcea, University of North Texas  
Dilek Hakkani-Tur, Microsoft  
Walter Daelemans, University of Antwerp  
Dan Roth, University of Illinois Urbana Champaign  
Alex Koller, University of Potsdam  
Ani Nenkova, University of Pennsylvania  
Jamie Henderson, XRCE  
Sadao Kurohashi, University of Kyoto  
Yuji Matsumoto, Nara Institute of S&T  
Heng Ji, CUNY  
Marie-Francine Moens, KU Leuven  
Hwee Tou Ng, NU Singapore

### **Program Committee:**

Abend Omri, Abney Steven, Abu-Jbara Amjad, Agarwal Apoorv, Agirre Eneko, Aguado-de-Cea Guadalupe, Ahrenberg Lars, Akkaya Cem, Alfonseca Enrique, Alishahi Afra, Allauzen Alexander, Altun Yasemin, Androutsopoulos Ion, Araki Masahiro, Artiles Javier, Artzi Yoav, Asahara Masayuki, Asher Nicholas, Atserias Batalla Jordi, Attardi Giuseppe, Ayan Necip Fazil

Baker Collin, Baldridge Jason, Baldwin Timothy, Banchs Rafael E., Banea Carmen, Bangalore Srinivas, Baroni Marco, Barrault Loïc, Barreiro Anabela, Basili Roberto, Bateman John, Bechet Frederic, Beigman Klebanov Beata, Bel Núria, Benajiba Yassine, Bender Emily M., Bendersky Michael, Benotti Luciana, Bergler Sabine, Besacier Laurent, Bethard Steven, Bicknell Clinton, Biemann Chris, Bikel Dan, Birch Alexandra, Bisazza Arianna, Blache Philippe, Bloodgood Michael, Bod Rens, Boitet Christian, Bojar Ondrej, Bond Francis, Bontcheva Kalina, Bordino Ilaria, Bosch Sonja, Boschee Elizabeth, Botha Jan, Bouma Gosse, Boye Johan, Boyer Kristy, Bracewell David, Branco António, Brants Thorsten, Brew Chris, Briscoe Ted, Bu Fan, Buitelaar Paul, Bunescu Razvan, Busemann Stephan, Byrne Bill, Byron Donna

Cabrio Elena, Cahill Aoife, Cahill Lynne, Callison-Burch Chris, Calzolari Nicoletta, Campbell Nick, Cancedda Nicola, Cao Hailong, Caragea Cornelia, Carberry Sandra, Cardenosa Jesus, Cardie Claire, Carl Michael, Carpuat Marine, Carreras Xavier, Carroll John, Casacuberta Francisco, Caselli Tommaso, Cassidy Steve, Cassidy Taylor, Celikyilmaz Asli, Cerisara Christophe, Chambers Nate, Chang Jason, Chang Kai-Wei, Chang Ming-Wei, Chang Jing-Shin, Chelba Ciprian, Chen Wenliang, Chen Zheng, Chen Wenliang, Chen John, Chen Boxing, Chen David, Cheng Pu-Jen, Cherry Colin, Chiang David, Choi Yejin, Choi Key-Sun, Christodoulopoulos Christos, Chrupala Grzegorz, Chu-Carroll Jennifer, Clark Stephen, Clark Peter, Cohn Trevor, Collier Nigel, Conroy John, Cook Paul, Coppola Bonaventura, Corazza Anna, Core Mark, Costa-jussà Marta R., Cristea Dan, Croce Danilo, Culotta Aron, da Cunha Iria

Daelemans Walter, Dagan Ido, Daille Beatrice, Danescu-Niculescu-Mizil Cristian, Dang Hoa Trang, Danlos Laurence, Das Dipanjan, de Gispert Adrià, De La Clergerie Eric, de Marneffe Marie-Catherine, de Melo Gerard, Declerck Thierry, Delmonte Rodolfo, Demberg Vera, DeNero John, Deng Hongbo, Denis Pascal, Deoras Anoop, DeVault David, Di Eugenio Barbara, Di Fabrizio Giuseppe, Diab Mona, Diaz de Ilarraza Arantza, Diligenti Michelangelo, Dinarelli Marco, Dipper Stefanie, Do Quang, Downey Doug, Dragut Eduard, Dreyer Markus, Du Jinhua, Duh Kevin, Dymetman Marc

Eberle Kurt, Eguchi Koji, Eisele Andreas, Elhadad Michael, Erk Katrin, Esuli Andrea, Evert Stefan

Fader Anthony, Fan James, Fang Hui, Favre Benoit, Fazly Afsaneh, Federico Marcello, Feldman Anna, Feldman Naomi, Fellbaum Christiane, Feng Junlan, Fernandez Raquel, Filippova Katja, Finch Andrew, Fišer Darja, Fleck Margaret, Forcada Mikel, Foster Jennifer, Foster George, Frank Stella, Frank Stefan L., Frank Anette, Fraser Alexander

Gabrilovich Evgeniy, Gaizauskas Robert, Galley Michel, Gamon Michael, Ganitkevitch Juri, Gao Jianfeng, Gardent Claire, Garrido Guillermo, Gatt Albert, Gavrilidou Maria, Georgila Kallirroi, Gesmundo Andrea, Gildea Daniel, Gill Alastair, Gillenwater Jennifer, Gillick Daniel, Girju Roxana, Giuliano Claudio, Gliozzo Alfio, Goh Chooi-Ling, Goldberg Yoav, Goldwasser Dan, Goldwater Sharon, Gonzalo Julio, Grau Brigitte, Green Nancy, Greene Stephan, Grefenstette Gregory, Grishman Ralph, Guo Jiafeng, Gupta Rahul, Gurevych Iryna, Gustafson Joakim, Guthrie Louise, Gutiérrez Yoan

Habash Nizar, Hachey Ben, Haddow Barry, Hahn Udo, Hall David, Harabagiu Sanda, Hardmeier Christian, Hashimoto Chikara, Hayashi Katsuhiko, He Xiaodong, He Zhongjun, Heid Uli, Heinz Jeffrey, Henderson John, Hendrickx Iris, Hermjakob Ulf, Hirst Graeme, Hoang Hieu, Hockenmaier Julia, Hoffart Johannes, Hopkins Mark, Horak Ales, Hori Chiori, Hoste Veronique, Hovy Eduard, Hsieh Shu-Kai, Hsu Wen-Lian, Huang Xuanjing, Huang Minlie, Huang Liang, Huang Chu-Ren, Huang Xuanjing, Huang Liang, Huang Fei, Hwang Mei-Yuh

Iglesias Gonzalo, Ikbal Shajith, Ilisei Iustina, Inkpen Diana, Isabelle Pierre, Isahara Hitoshi, Ittycheriah Abe

Jaeger T. Florian, Jagarlamudi Jagadeesh, Jiampojarn Sittichai, Jiang Xing, Jiang Wenbin, Jiang Jing, Johansson Richard, Johnson Mark, Johnson Howard, Jurgens David

Kageura Kyo, Kan Min-Yen, Kanoulas Evangelos, Kanzaki Kyoko, Kawahara Daisuke, Keizer Simon, Kelleher John, Kempe Andre, Keshtkar Fazel, Khadivi Shahram, Kilgarriff Adam, King Tracy Holloway, Kit Chunyu, Knight Kevin, Koehn Philipp, Koeling Rob, Kolomiyets Oleksandr, Komatani Kazunori, Kondrak Grzegorz, Kong Fang, Kopp Stefan, Koppel Moshe, Kordoni Valia, Kozareva Zornitsa, Kozhevnikov Mikhail, Kraemer Emiel, Kremer Gerhard, Kudo Taku, Kuhlmann Marco, Kuhn Roland, Kumar Shankar, Kundu Gourab, Kurland Oren

Lam Wai, Lamar Michael, Lambert Patrik, Langlais Phillippe, Lapalme Guy, Lapata Mirella, Laws Florian, Leacock Claudia, Lee Yoong Keok, Lee Lin-shan, Lee Gary Geunbae, Lee Yoong Keok, Lee Sungjin, Lee John, Lefevre Fabrice, Lemon Oliver, Lenci Alessandro, Leong Ben, Leusch Gregor, Levenberg Abby, Levy Roger, Li Linlin, Li Fangtao, Li Yan, Li Haibo, Li Wenjie, Li Shoushan, Li Qi, Li Haizhou, Li Tao, Liao Shasha, Lin Dekang, Lin Ziheng, Lin Hui, Lin Ziheng, Lin Thomas, Litvak Marina, Liu Yang, Liu Bing, Liu Qun, Liu Ting, Liu Fei, Liu Zhiyuan, Liu Yiqun, Liu Chang, Liu Zhiyuan, Liu Jingjing, Liu Yiqun, Ljubešić Nikola, Lloret Elena, Lopez Adam, Lopez-Cozar Ramon, Louis Annie, Lu Wei, Lu Xiaofei, Lu Yue, Luca Dini, Luo Xiaoqiang, Lv Yajuan

Ma Yanjun, Macherey Wolfgang, Macherey Klaus, Madnani Nitin, Maegaard Bente, Magnini Bernardo, Maier Andreas, Manandhar Suresh, Marcu Daniel, Markantonatou Stella, Markert Katja, Marsi Erwin, Martin James H., Martinez David, Mason Rebecca, Matsubara Shigeki, Matsumoto

Yuji, Matsuzaki Takuya, Mauro Cettolo, Mauser Arne, May Jon, Mayfield James, Maynard Diana, McCarthy Diana, McClosky David, McCoy Kathy, McCrae John Philip, McNamee Paul, Meij Edgar, Mejova Yelena, Mellish Chris, Merlo Paola, Metze Florian, Metzler Donald, Meyers Adam, Mi Haitao, Mihalcea Rada, Miltsakaki Eleni, Minkov Einat, Mitchell Margaret, Miyao Yusuke, Mochihashi Daichi, Moens Marie-Francine, Mohammad Saif, Moilanen Karo, Monson Christian, Montes Manuel, Monz Christof, Moon Taesun, Moore Robert, Morante Roser, Morarescu Paul, Mueller Thomas, Munteanu Dragos, Murawaki Yugo, Muresan Smaranda, Myaeng Sung-Hyon, Mylonakis Markos

Nakagawa Tetsuji, Nakano Mikio, Nakazawa Toshiaki, Nakov Preslav, Naradowsky Jason, Naseem Tahira, Nastase Vivi, Navarro Borja, Navigli Roberto, Nazarenko Adeline, Nederhof Mark-Jan, Negri Matteo, Nenkova Ani, Neubig Graham, Neumann Guenter, Ng Vincent, Ngai Grace, Nguyen ThuyLinh, Nivre Joakim, Nowson Scott

Och Franz, Odijk Jan, Oflazer Kemal, Oh Jong-Hoon, Okazaki Naoaki, Oltramari Alessandro, Orasan Constantin, Osborne Miles, Osenova Petya, Ott Myle, Ovesdotter Alm Cecilia

Padó Sebastian, Palmer Martha, Palmer Alexis, Pang Bo, Pantel Patrick, Paraboni Ivandre, Pardo Thiago, Paris Cecile, Paroubek Patrick, Patwardhan Siddharth, Paul Michael, Paulik Matthias, Pearl Lisa, Pedersen Ted, Pedersen Bolette, Pedersen Ted, Peñas Anselmo, Penn Gerald, Perez-Rosas Veronica, Peters Wim, Petrov Slav, Petrovic Sasa, Piasecki Maciej, Pighin Daniele, Pinkal Manfred, Piperidis Stelios, Piskorski Jakub, Pitler Emily, Plank Barbara, Ponzetto Simone Paolo, Popescu Octavian, Popescu-Belis Andrei, Popović Maja, Potts Christopher, Pradhan Sameer, Prager John, Prasad Rashmi, Prószyński Gábor, Pulman Stephen, Punyakanok Vasin, Purver Matthew, Pustejovsky James

Qazvinian Vahed, Qian Xian, Qu Shaolin, Quarteroni Silvia, Quattoni Ariadna, Quirk Chris

Raaijmakers Stephan, Rahman Altaf, Rambow Owen, Rao Delip, Rappoport Ari, Ravi Sujith, Rayner Manny, Recasens Marta, Regneri Michaela, Reichart Roi, Reitter David, Resnik Philip, Riccardi Giuseppe, Riedel Sebastian, Riesa Jason, Rieser Verena, Riezler Stefan, Rigau German, Ringgaard Michael, Ritter Alan, Roark Brian, Rodriguez Horacio, Rohde Hannah, Rosenberg Andrew, Rosso Paolo, Rozovskaya Alla, Rus Vasile, Rusu Delia

Sagae Kenji, Sahakian Sam, Saint-Dizier Patrick, Samdani Rajhans, Sammons Mark, Sangal Rajeev, Saraclar Murat, Sarkar Anoop, Sassano Manabu, Satta Giorgio, Saurí Roser, Scaiano Martin, Schlangen David, Schmid Helmut, Schneider Nathan, Schulte im Walde Sabine, Schwenk Holger, Segond Frederique, Seki Yohei, Sekine Satoshi, Senellart Jean, Setiawan Hendra, Severyn Aliaksei, Shanker Vijay, Sharma Dipti, Sharoff Serge, Shi Shuming, Shi Xiaodong, Shi Shuming, Shutova Ekaterina, Si Xiance, Sidner Candace, Silva Mario J., Sima'an Khalil, Simard Michel, Skantze Gabriel, Small Kevin, Smith Noah A., Smith Nathaniel, Smrz Pavel, Smrz Pavel, Šnajder Jan, Snyder Benjamin, Sjøgaard Anders, Solorio Tamar, Somasundaran Swapna, Song Yangqiu, Spotovský Valentin, Sporleder Caroline, Sprugnoli Rachele, Srikumar Vivek, Stede Manfred, Steedman Mark, Steinberger Ralf, Stevenson Mark, Stone Matthew, Stoyanov Veselin, Strube Michael, Strzalkowski Tomek, Stymne Sara, Su Keh-Yih, Su Jian, Sun Ang, Surdeanu Mihai, Suzuki Hisami, Schwartz Roy, Szpakowicz Stan, Szpektor Idan

Täckström Oscar, Takamura Hiroya, Talukdar Partha, Tatu Marta, Taylor Sarah, Tenbrink Thora, Thater Stefan, Tiedemann Jörg, Tillmann Christoph, Titov Ivan, Toivonen Hannu, Tokunaga Takenobu,

Tonelli Sara, Toutanova Kristina, Tsarfaty Reut, Tsochantaridis Ioannis, Tsujii Jun'ichi, Tsukada Hajime, Tsuruoka Yoshimasa, Tufis Dan, Tur Gokhan, Turney Peter, Tymoshenko Kateryna

Uchimoto Kiyotaka, Udupa Raghavendra, Uryupina Olga, Utiyama Masao

Valitutti Alessandro, van den Bosch Antal, van der Plas Lonneke, Van Durme Benjamin, van Genabith Josef, Van Huyssteen Gerhard, van Noord Gertjan, Vandeghinste Vincent, Veale Tony, Velardi Paola, Verhagen Marc, Vetulani Zygmunt, Viethen Jette, Vieu Laure, Vilar David, Villavicencio Aline, Virpioja Sami, Voorhees Ellen, Vossen Piek, Vulić Ivan

Walker Marilyn, Wan Stephen, Wan Xiaojun, Wang Lu, Wang Chi, Wang Jun, Wang Haifeng, Wang Mengqiu, Wang Quan, Wang Wen, Ward Nigel, Washtell Justin, Watanabe Taro, Webber Bonnie, Wei Furu, Welty Chris, Wen Zhen, Wen Ji-Rong, Wen Zhen, Wicentowski Rich, Widdows Dominic, Wiebe Jan, Williams Jason, Wilson Theresa, Wintner Shuly, Wong Kam-Fai, Woodsend Kristian, Wooters Chuck, Wu Xianchao

Xiao Tong, Xiong Deyi, Xu Wei, Xu Jun, Xue Nianwen, Xue Xiaobing

Yan Rui, Yang Muyun, Yang Bishan, Yangarber Roman, Yano Tae, Yao Limin, Yates Alexander, Yatskar Mark, Yih Wen-tau, Yli-Jyrä Anssi, Yu Bei, Yvon François

Zabokrtsky Zdenek, Zanzotto Fabio Massimo, Zens Richard, Zettlemoyer Luke, Zeyrek Deniz, Zhang Yue, Zhang Min, Zhang Ruiqiang, Zhang Hao, Zhang Yue, Zhang Hui, Zhang Yi, Zhang Joy Ying, Zhanyi Liu, Zhao Hai, Zhao Tiejun, Zhao Jun, Zhao Shiqi, Zheng Jing, Zhou Guodong, Zhou Ming, Zhou Ke, Zhou Guodong, Zhou Ming, Zhou Guodong, Zhu Jingbo, Zhu Xiaodan, Zock Michael, Zukerman Ingrid, Zweigenbaum Pierre.

## Invited Talk

### When parsing makes things worse: An eye-tracking study of English compounds

Harald Baayen

Seminar für Sprachwissenschaft, Eberhard Karls University, Tuebingen

### Abstract

Compounds differ in the degree to which they are semantically compositional (compare, e.g., "carwash", "handbag", "beefcake" and "humbug"). Since even relatively transparent compounds such as "carwash" may leave the uninitiated reader with uncertainty about the intended meaning (soap for washing cars? a place where you can get your car washed?), an efficient way of retrieving the meaning of a compound is to use the compound's form as an access key for its meaning.

However, in psychology, the view has become popular that at the earliest stage of lexical processing in reading, a morpho-orthographic decomposition into morphemes would necessarily take place. Theorists ascribing to obligatory decomposition appear to have some hash coding scheme in mind, with the constituents providing entry points to a form of table look-up (e.g., Taft & Forster, 1976).

Leaving aside the question of whether such a hash coding scheme would be computationally efficient as well as the question how the putative morpho-orthographic representations would be learned, my presentation focuses on the details of lexical processing as revealed by an eye-tracking study of the reading of English compounds in sentences.

A careful examination of the eye-tracking record with generalized additive modeling (Wood, 2006), combined with computational modeling using naive discrimination learning (Baayen, Milin, Filipovic, Hendrix, & Marelli, 2011) revealed that how far the eye moved into the compound is co-determined by the compound's lexical distributional properties, including the cosine similarity of the compound and its head in document vector space (as measured with latent semantic analysis, Landauer & Dumais, 1997). This indicates that compound processing is initiated already while the eye is fixating on the preceding word, and that even before the eye has landed on the compound, processes discriminating the meaning of the compound from the meaning of its head have already come into play.

Once the eye lands on the compound, two very different reading signatures emerge, which critically depend on the letter trigrams spanning the morpheme boundary (e.g., "ndb" and "dba" in "handbag"). From a discrimination learning perspective, these boundary trigrams provide the crucial (and only) orthographic cues for the compound's (idiosyncratic) meaning. If the boundary trigrams are sufficiently strongly associated with the compound's meaning, and if the eye lands early enough in the word, a single fixation suffices. Within 240 ms (of which 80 ms involve planning the next saccade) the compound's meaning is discriminated well enough to proceed to the next word.

However, when the boundary trigrams are only weakly associated with the compound's meaning, multiple fixations become necessary. In this case, without the availability of the critical orthographic cues, the eye-tracking record bears witness to the cognitive system engaging not only bottom-up processes from form to meaning, but also top-down guessing processes that are informed by the a-priori probability of the head and the cosine similarities of the compound and its constituents in semantic vector space.

These results challenge theories positing obligatory decomposition with hash coding, as hash coding predicts insensitivity to semantic transparency, contrary to fact. Our results also challenge theories positing blind look-up based on compounds' orthographic forms. Although this might be computationally efficient, the eye can't help seeing parts of the whole. In summary, reality is much more complex, with deep pre-arrival parafoveal processing followed by either efficient discrimination driven by the boundary

trigrams (within 140 ms), or by an inefficient decompositional process (requiring an additional 200 ms) that seeks to make sense of the conjunction of head and modifier.

### **References**

Baayen, R. H., Kuperman, V., Shaoul, C., Milin, P., Kliegl, R. & Ramscar, M. (submitted), Decomposition makes things worse. A discrimination learning approach to the time course of understanding compounds in reading.

Baayen, R. H., Milin, P., Filipovic Durdjevic, D., Hendrix, P. & Marelli, M. (2011), An amorphous model for morphological processing in visual comprehension based on naive discriminative learning, *Psychological Review*, 118, 3, 438-481.

Landauer, T.K. & Dumais, S.T. (1997), A Solution to Plato's Problem: The Latent Semantic Analysis theory of acquisition, induction and representation of knowledge, *Psychological Review*, 104, 2, 211-240.

Taft, M. & Forster, K. I. (1976), Lexical Storage and Retrieval of Polymorphemic and Polysyllabic Words, *Journal of Verbal Learning and Verbal Behavior*, 15, 607-620.

Wood, S. N. (2006), *Generalized Additive Models*, Chapman & Hall/CRC, New York.

## **Invited Talk**

### **The Natural Language Interface of Graph Search**

**Lars Rasmussen**

Facebook Inc

### **Abstract**

The backbone of the Facebook social network service is an enormous graph representing hundreds of types of nodes and thousands of types of edges. Among these nodes are over 1 billion users and 250 billion photos. The edges connecting these nodes have exceeded 1 trillion and continue to grow at an incredible rate. Retrieving information from such a graph has been a formidable and exciting task. Now it is possible for you to find, in an aggregated manner, restaurants in a city that your friends have visited, or photos of people who have attended college with you, and explore many other nuanced connections between the nodes and edges in our graph given that such information is visible to you.

Graph Search Beta, launched early this year, is a personalized semantic search engine that allows users to express their intent in natural language. It seeks answers through the traversal of relevant graph edges and ranks results by various signals extracted from our data. You can find “tv shows liked by people who study linguistics“ by issuing this query verbatim and, for the entertainment value, compare the results with “tv shows liked by people who study computer science“. Our system is built to be robust to many varied inputs, such as grammatically incorrect user queries or traditional keyword searches. Our query suggestions are always constructed in natural language, expressing the precise intention interpreted by our system. This means users would know in advance whether the system has correctly understood their intent before selecting any suggestion. The system also assists users with auto-completions, demonstrating what kinds of queries it can understand.

The development of the natural language interface encountered an array of challenging problems. The grammar structure needed to incorporate semantic information in order to translate an unstructured query into a structured semantic function, and also use syntactic information to return grammatically meaningful suggestions. The system required not only the recognition of entities in a query, but also the resolution of entities to database entries based on proximity of the entity and user nodes. Semantic parsing aimed to rank potential semantics including those that may match the immediate purpose of the query along with other refinements of the original intent. The ambiguous nature of natural language led us to consider how to interpret certain queries in the most sensible way. The need for speed demanded state-of-the-art parsing algorithms tailored for our system. In this talk, I will introduce the audience to Graph Search Beta, share our experience in developing the technical components of the natural language interface, and bring up topics that may be of interesting research value to the NLP community.

## **Invited Talk**

### **Individual Differences in Language and Executive Processes: How the Brain Keeps Track of Variables**

**Chantel S. Prat**

University of Washington

#### **Abstract**

Language comprehension is a complex cognitive process which requires tracking and integrating multiple variables. Thus, it is not surprising that language abilities (e.g., reading comprehension) vary widely even in the college population, and that language and general cognitive abilities (e.g., working memory capacity) co-vary. Although it has been widely accepted that improvements in general cognitive abilities enable (or give rise to) increased linguistic skills, the fact that individuals who develop bilingually outperform monolinguals in tests of executive functioning provides evidence of a situation in which a particular language experience gives rise to improvements in general cognitive processes. In this talk, I will describe two converging lines of research investigating individual differences in working memory capacity and reading ability in monolinguals and improved executive functioning in bilinguals. Results from these investigations suggest that the functioning of the fronto-striatal loops can explain the relation between language and non-linguistic executive functioning in both populations. I then discuss evidence suggesting that this system may function to track and route “variables” into prefrontal control structures.

## Table of Contents

<i>A Shift-Reduce Parsing Algorithm for Phrase-based String-to-Dependency Translation</i> Yang Liu .....	1
<i>Integrating Translation Memory into Phrase-Based Machine Translation during Decoding</i> Kun Wang, Chengqing Zong and Keh-Yih Su .....	11
<i>Training Nondeficient Variants of IBM-3 and IBM-4 for Word Alignment</i> Thomas Schoenemann .....	22
<i>Modelling Annotator Bias with Multi-task Gaussian Processes: An Application to Machine Translation Quality Estimation</i> Trevor Cohn and Lucia Specia .....	32
<i>Smoothed marginal distribution constraints for language modeling</i> Brian Roark, Cyril Allauzen and Michael Riley .....	43
<i>Grounded Language Learning from Video Described with Sentences</i> Haonan Yu and Jeffrey Mark Siskind .....	53
<i>Plurality, Negation, and Quantification: Towards Comprehensive Quantifier Scope Disambiguation</i> Mehdi Manshadi, Daniel Gildea and James Allen .....	64
<i>Joint Event Extraction via Structured Prediction with Global Features</i> Qi Li, Heng Ji and Liang Huang .....	73
<i>Language-Independent Discriminative Parsing of Temporal Expressions</i> Gabor Angeli and Jakob Uszkoreit .....	83
<i>Graph-based Local Coherence Modeling</i> Camille Guinaudeau and Michael Strube .....	93
<i>Recognizing Rare Social Phenomena in Conversation: Empowerment Detection in Support Group Chatrooms</i> Elijah Mayfield, David Adamson and Carolyn Penstein Rosé .....	104
<i>Decentralized Entity-Level Modeling for Coreference Resolution</i> Greg Durrett, David Hall and Dan Klein .....	114
<i>Chinese Parsing Exploiting Characters</i> Meishan Zhang, Yue Zhang, Wanxiang Che and Ting Liu .....	125
<i>A Transition-Based Dependency Parser Using a Dynamic Parsing Strategy</i> Francesco Sartorio, Giorgio Satta and Joakim Nivre .....	135
<i>General binarization for parsing and translation</i> Matthias Büchse, Alexander Koller and Heiko Vogler .....	145
<i>Distortion Model Considering Rich Context for Statistical Machine Translation</i> Isao Goto, Masao Utiyama, Eiichiro Sumita, Akihiro Tamura and Sadao Kurohashi .....	155
<i>Word Alignment Modeling with Context Dependent Deep Neural Network</i> Nan Yang, Shujie Liu, Mu Li, Ming Zhou and Nenghai Yu .....	166

<i>Microblogs as Parallel Corpora</i>	
Wang Ling, Guang Xiang, Chris Dyer, Alan Black and Isabel Trancoso .....	176
<i>Improved Bayesian Logistic Supervised Topic Models with Data Augmentation</i>	
Jun Zhu, Xun Zheng and Bo Zhang .....	187
<i>Fast and Robust Compressive Summarization with Dual Decomposition and Multi-Task Learning</i>	
Miguel Almeida and Andre Martins .....	196
<i>Unsupervised Transcription of Historical Documents</i>	
Taylor Berg-Kirkpatrick, Greg Durrett and Dan Klein .....	207
<i>Adapting Discriminative Reranking to Grounded Language Learning</i>	
Joohyun Kim and Raymond Mooney .....	218
<i>Universal Conceptual Cognitive Annotation (UCCA)</i>	
Omri Abend and Ari Rappoport .....	228
<i>Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media</i>	
Weiwei Guo, Hao Li, Heng Ji and Mona Diab .....	239
<i>A computational approach to politeness with application to social factors</i>	
Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec and Christopher Potts .....	250
<i>Modeling Thesis Clarity in Student Essays</i>	
Isaac Persing and Vincent Ng .....	260
<i>Translating Italian connectives into Italian Sign Language</i>	
Camillo Lugaresi and Barbara Di Eugenio .....	270
<i>Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing</i>	
David Mareček and Milan Straka .....	281
<i>Transfer Learning for Constituency-Based Grammars</i>	
Yuan Zhang, Regina Barzilay and Amir Globerson .....	291
<i>A Context Free TAG Variant</i>	
Ben Swanson, Elif Yamangil, Eugene Charniak and Stuart Shieber .....	302
<i>Fast and Adaptive Online Training of Feature-Rich Translation Models</i>	
Spence Green, Sida Wang, Daniel Cer and Christopher D. Manning .....	311
<i>Advancements in Reordering Models for Statistical Machine Translation</i>	
Minwei Feng, Jan-Thorsten Peter and Hermann Ney .....	322
<i>A Markov Model of Machine Translation using Non-parametric Bayesian Inference</i>	
Yang Feng and Trevor Cohn .....	333
<i>Scaling Semi-supervised Naive Bayes with Feature Marginals</i>	
Michael Lucas and Doug Downey .....	343
<i>Learning Latent Personas of Film Characters</i>	
David Bamman, Brendan O'Connor and Noah A. Smith .....	352

<i>Scalable Decipherment for Machine Translation via Hash Sampling</i> Sujith Ravi .....	362
<i>Automatic Interpretation of the English Possessive</i> Stephen Tratz and Eduard Hovy .....	372
<i>Is a 204 cm Man Tall or Small ? Acquisition of Numerical Common Sense from the Web</i> Katsuma Narisawa, Yotaro Watanabe, Junta Mizuno, Naoaki Okazaki and Kentaro Inui .....	382
<i>Probabilistic Domain Modelling With Contextualized Distributional Semantic Vectors</i> Jackie Chi Kit Cheung and Gerald Penn .....	392
<i>Extracting bilingual terminologies from comparable corpora</i> Ahmet Aker, Monica Paramita and Rob Gaizauskas .....	402
<i>The Haves and the Have-Nots: Leveraging Unlabelled Corpora for Sentiment Analysis</i> Kashyap Popat, Balamurali A.R, Pushpak Bhattacharyya and Gholamreza Haffari .....	412
<i>Large-scale Semantic Parsing via Schema Matching and Lexicon Extension</i> Qingqing Cai and Alexander Yates .....	423
<i>Fast and Accurate Shift-Reduce Constituent Parsing</i> Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang and Jingbo Zhu .....	434
<i>Nonconvex Global Optimization for Latent-Variable Models</i> Matthew R. Gormley and Jason Eisner .....	444
<i>Parsing with Compositional Vector Grammars</i> Richard Socher, John Bauer, Christopher D. Manning and Ng Andrew Y. ....	455
<i>Discriminative state tracking for spoken dialog systems</i> Angeliki Metallinou, Dan Bohus and Jason Williams .....	466
<i>Leveraging Synthetic Discourse Data via Multi-task Learning for Implicit Discourse Relation Recognition</i> Man Lan, Yu Xu and Zhengyu Niu .....	476
<i>Combining Intra- and Multi-sentential Rhetorical Parsing for Document-level Discourse Analysis</i> Shafiq Joty, Giuseppe Carenini, Raymond Ng and Yashar Mehdad .....	486
<i>Improving pairwise coreference models through feature space hierarchy learning</i> Emmanuel Lassalle and Pascal Denis .....	497
<i>Feature-Based Selection of Dependency Paths in Ad Hoc Information Retrieval</i> K. Tamsin Maxwell, Jon Oberlander and W. Bruce Croft .....	507
<i>Coordination Structures in Dependency Treebanks</i> Martin Popel, David Mareček, Jan Štěpánek, Daniel Zeman and Zdeněk Žabokrtský .....	517
<i>GlossBoot: Bootstrapping Multilingual Domain Glossaries from the Web</i> Flavio De Benedictis, Stefano Faralli and Roberto Navigli .....	528
<i>Collective Annotation of Linguistic Resources: Basic Principles and a Formal Model</i> Ulle Endriss and Raquel Fernández .....	539

<i>ParGramBank: The ParGram Parallel Treebank</i>	
Sebastian Sulger, Miriam Butt, Tracy Holloway King, Paul Meurer, Tibor Laczkó, György Rákosi, Cheikh Bamba Dione, Helge Dyvik, Victoria Rosén, Koenraad De Smedt, Agnieszka Patejuk, Ozlem Cetinoglu, I Wayan Arka and Meladel Mistica . . . . .	550
<i>Identifying Bad Semantic Neighbors for Improving Distributional Thesauri</i>	
Olivier Ferret . . . . .	561
<i>Models of Semantic Representation with Visual Attributes</i>	
Carina Silberer, Vittorio Ferrari and Mirella Lapata . . . . .	572
<i>Real-World Semi-Supervised Learning of POS-Taggers for Low-Resource Languages</i>	
Dan Garrette, Jason Mielens and Jason Baldridge . . . . .	583
<i>Using subcategorization knowledge to improve case prediction for translation to German</i>	
Marion Weller, Alexander Fraser and Sabine Schulte im Walde . . . . .	593
<i>Name-aware Machine Translation</i>	
Haibo Li, Jing Zheng, Heng Ji, Qi Li and Wen Wang . . . . .	604
<i>Decipherment Complexity in 1:1 Substitution Ciphers</i>	
Malte Nuhn and Hermann Ney . . . . .	615
<i>Non-Monotonic Sentence Alignment via Semisupervised Learning</i>	
Xiaojun Quan, Chunyu Kit and Yan Song . . . . .	622
<i>Bootstrapping Entity Translation on Weakly Comparable Corpora</i>	
Taesung Lee and Seung-won Hwang . . . . .	631
<i>Transfer Learning Based Cross-lingual Knowledge Extraction for Wikipedia</i>	
Zhigang Wang, Zhixing Li, Juanzi Li, Jie Tang and Jeff Z. Pan . . . . .	641
<i>Bridging Languages through Etymology: The case of cross language text categorization</i>	
Vivi Nastase and Carlo Strapparava . . . . .	651
<i>Creating Similarity: Lateral Thinking for Vertical Similarity Judgments</i>	
Tony Veale and Guofu Li . . . . .	660
<i>Discovering User Interactions in Ideological Discussions</i>	
Arjun Mukherjee and Bing Liu . . . . .	671
<i>Multilingual Affect Polarity and Valence Prediction in Metaphor-Rich Texts</i>	
Zornitsa Kozareva . . . . .	682
<i>Large tagset labeling using Feed Forward Neural Networks. Case study on Romanian Language</i>	
Tiberiu Boros, Radu Ion and Dan Tufis . . . . .	692
<i>Learning to lemmatise Polish noun phrases</i>	
Adam Radziszewski . . . . .	701
<i>Using Conceptual Class Attributes to Characterize Social Media Users</i>	
Shane Bergsma and Benjamin Van Durme . . . . .	710
<i>The Impact of Topic Bias on Quality Flaw Prediction in Wikipedia</i>	
Oliver Ferschke, Iryna Gurevych and Marc Rittberger . . . . .	721

<i>Mining Informal Language from Chinese Microtext: Joint Word Recognition and Segmentation</i> Aobo Wang and Min-Yen Kan .....	731
<i>Generating Synthetic Comparable Questions for News Articles</i> Oleg Rokhlenko and Idan Szpektor .....	742
<i>Punctuation Prediction with Transition-based Parsing</i> Dongdong Zhang, Shuangzhi Wu, Nan Yang and Mu Li .....	752
<i>Discriminative Learning with Natural Annotations: Word Segmentation as a Case Study</i> Wenbin Jiang, Meng Sun, Yajuan Lü, Yating Yang and Qun Liu .....	761
<i>Graph-based Semi-Supervised Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging</i> Xiaodong Zeng, Derek F. Wong, Lidia S. Chao and Isabel Trancoso .....	770
<i>An Infinite Hierarchical Bayesian Model of Phrasal Translation</i> Trevor Cohn and Gholamreza Haffari .....	780
<i>Additive Neural Networks for Statistical Machine Translation</i> Jiemao Liu, Taro Watanabe, Eiichiro Sumita and Tiejun Zhao .....	791
<i>Hierarchical Phrase Table Combination for Machine Translation</i> Conghui Zhu, Taro Watanabe, Eiichiro Sumita and Tiejun Zhao .....	802
<i>Shallow Local Multi-Bottom-up Tree Transducers in Statistical Machine Translation</i> Fabienne Braune, Nina Seemann, Daniel Quernheim and Andreas Maletti .....	811
<i>Enlisting the Ghost: Modeling Empty Categories for Machine Translation</i> Bing Xiang, Xiaoqiang Luo and Bowen Zhou .....	822
<i>A Multi-Domain Translation Model Framework for Statistical Machine Translation</i> Rico Sennrich, Holger Schwenk and Walid Aransa .....	832
<i>Part-of-Speech Induction in Dependency Trees for Statistical Machine Translation</i> Akihiro Tamura, Taro Watanabe, Eiichiro Sumita, Hiroya Takamura and Manabu Okumura ...	841
<i>Statistical Machine Translation Improves Question Retrieval in Community Question Answering via Matrix Factorization</i> Guangyou Zhou, Fang Liu, Yang Liu, Shizhu He and Jun Zhao .....	852
<i>Improved Lexical Acquisition through DPP-based Verb Clustering</i> Roi Reichart and Anna Korhonen .....	862
<i>Semantic Frames to Predict Stock Price Movement</i> Boyi Xie, Rebecca J. Passonneau, Leon Wu and Germán G. Creamer .....	873
<i>Density Maximization in Context-Sense Metric Space for All-words WSD</i> Koichi Tanigaki, Mitsuteru Shiba, Tatsuji Munaka and Yoshinori Sagisaka .....	884
<i>The Role of Syntax in Vector Space Models of Compositional Semantics</i> Karl Moritz Hermann and Phil Blunsom .....	894
<i>Margin-based Decomposed Amortized Inference</i> Gourab Kundu, Vivek Srikumar and Dan Roth .....	905

<i>Semi-Supervised Semantic Tagging of Conversational Understanding using Markov Topic Regression</i> Asli Celikyilmaz, Dilek Hakkani-Tur, Gokhan Tur and Ruhi Sarikaya .....	914
<i>Parsing Graphs with Hyperedge Replacement Grammars</i> David Chiang, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, Bevan Jones and Kevin Knight	924
<i>Grounded Unsupervised Semantic Parsing</i> Hoifung Poon .....	933
<i>Automatic detection of deception in child-produced speech using syntactic complexity features</i> Maria Yancheva and Frank Rudzicz .....	944
<i>Sentiment Relevance</i> Christian Scheible and Hinrich Schütze .....	954
<i>Predicting and Eliciting Addressee’s Emotion in Online Dialogue</i> Takayuki Hasegawa, Nobuhiro Kaji, Naoki Yoshinaga and Masashi Toyoda .....	964
<i>Utterance-Level Multimodal Sentiment Analysis</i> Veronica Perez-Rosas, Rada Mihalcea and Louis-Philippe Morency .....	973
<i>Probabilistic Sense Sentiment Similarity through Hidden Emotions</i> Mitra Mohtarami, Man Lan and Chew Lim Tan .....	983
<i>A user-centric model of voting intention from Social Media</i> Vasileios Lampos, Daniel Preoŧiuc-Pietro and Trevor Cohn .....	993
<i>Using Supervised Bigram-based ILP for Extractive Summarization</i> Chen Li, Xian Qian and Yang Liu .....	1004
<i>Summarization Through Submodularity and Dispersion</i> Anirban Dasgupta, Ravi Kumar and Sujith Ravi .....	1014
<i>Subtree Extractive Summarization via Submodular Maximization</i> Hajime Morita, Ryohei Sasano, Hiroya Takamura and Manabu Okumura .....	1023
<i>The effect of non-tightness on Bayesian estimation of PCFGs</i> Shay B. Cohen and Mark Johnson .....	1033
<i>Integrating Multiple Dependency Corpora for Inducing Wide-coverage Japanese CCG Resources</i> Sumire Uematsu, Takuya Matsuzaki, Hiroki Hanaoka, Yusuke Miyao and Hideki Mima .....	1042
<i>Transition-based Dependency Parsing with Selectional Branching</i> Jinho D. Choi and Andrew McCallum .....	1052
<i>Bilingually-Guided Monolingual Dependency Grammar Induction</i> Kai Liu, Yajuan Lü, Wenbin Jiang and Qun Liu .....	1063
<i>Joint Word Alignment and Bilingual Named Entity Recognition Using Dual Decomposition</i> Mengqiu Wang, Wanxiang Che and Christopher D. Manning .....	1073
<i>Resolving Entity Morphs in Censored Data</i> Hongzhao Huang, Zhen Wen, Dian Yu, Heng Ji, Yizhou Sun, Jiawei Han and He Li .....	1083

<i>Learning to Extract International Relations from Political Context</i> Brendan O'Connor, Brandon M. Stewart and Noah A. Smith .....	1094
<i>Graph Propagation for Paraphrasing Out-of-Vocabulary Words in Statistical Machine Translation</i> Majid Razmara, Maryam Siahbani, Reza Haffari and Anoop Sarkar .....	1105
<i>Online Relative Margin Maximization for Statistical Machine Translation</i> Vladimir Eidelman, Yuval Marton and Philip Resnik .....	1116
<i>Handling Ambiguities of Bilingual Predicate-Argument Structures for Statistical Machine Translation</i> Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong .....	1127
<i>Reconstructing an Indo-European Family Tree from Non-native English Texts</i> Ryo Nagata and Edward Whittaker .....	1137
<i>Word Association Profiles and their Use for Automated Scoring of Essays</i> Beata Beigman Klebanov and Michael Flor .....	1148
<i>Adaptive Parser-Centric Text Normalization</i> Congle Zhang, Tyler Baldwin, Howard Ho, Benny Kimelfeld and Yunyao Li .....	1159
<i>A Random Walk Approach to Selectional Preferences Based on Preference Ranking and Propagation</i> Zhenhua Tian, Hengheng Xiang, Ziqi Liu and Qinghua Zheng .....	1169
<i>ImpAr: A Deterministic Algorithm for Implicit Semantic Role Labelling</i> Egoitz Laparra and German Rigau .....	1180
<i>Cross-lingual Transfer of Semantic Role Labeling Models</i> Mikhail Kozhevnikov and Ivan Titov .....	1190
<i>DERivBase: Inducing and Evaluating a Derivational Morphology Resource for German</i> Britta Zeller, Jan Šnajder and Sebastian Padó .....	1201
<i>Crowdsourcing Interaction Logs to Understand Text Reuse from the Web</i> Martin Potthast, Matthias Hagen, Michael Völske and Benno Stein .....	1212
<i>SPred: Large-scale Harvesting of Semantic Predicates</i> Tiziano Flati and Roberto Navigli .....	1222
<i>Towards Robust Abstractive Multi-Document Summarization: A Caseframe Analysis of Centrality and Domain</i> Jackie Chi Kit Cheung and Gerald Penn .....	1233
<i>HEADY: News headline abstraction through event pattern clustering</i> Enrique Alfonseca, Daniele Pighin and Guillermo Garrido .....	1243
<i>Conditional Random Fields for Responsive Surface Realisation using Global Features</i> Nina Dethlefs, Helen Hastie, Heriberto Cuayáhuitl and Oliver Lemon .....	1254
<i>Two-Neighbor Orientation Model with Cross-Boundary Global Contexts</i> Hendra Setiawan, Bowen Zhou, Bing Xiang and Libin Shen .....	1264
<i>Cut the noise: Mutually reinforcing reordering and alignments for improved machine translation</i> Karthik Visweswariah, Mitesh M. Khapra and Ananthakrishnan Ramanathan .....	1275

<i>Vector Space Model for Adaptation in Statistical Machine Translation</i> Boxing Chen, Roland Kuhn and George Foster .....	1285
<i>From Natural Language Specifications to Program Input Parsers</i> Tao Lei, Fan Long, Regina Barzilay and Martin Rinard .....	1294
<i>Entity Linking for Tweets</i> Xiaohua Liu, Yitong Li, Haocheng Wu, Ming Zhou, Furu Wei and Yi Lu .....	1304
<i>Identification of Speakers in Novels</i> Hua He, Denilson Barbosa and Grzegorz Kondrak .....	1312
<i>Language Acquisition and Probabilistic Models: keeping it simple</i> Aline Villavicencio, Marco Idiart, Robert Berwick and Igor Malioutov .....	1321
<i>A Two Level Model for Context Sensitive Inference Rules</i> Oren Melamud, Jonathan Berant, Ido Dagan, Jacob Goldberger and Idan Szpektor .....	1331
<i>Align, Disambiguate and Walk: A Unified Approach for Measuring Semantic Similarity</i> Mohammad Taher Pilehvar, David Jurgens and Roberto Navigli .....	1341
<i>Linking and Extending an Open Multilingual Wordnet</i> Francis Bond and Ryan Foster .....	1352
<i>FrameNet on the Way to Babel: Creating a Bilingual FrameNet Using Wiktionary as Interlingual Connection</i> Silvana Hartmann and Iryna Gurevych .....	1363
<i>Dirt Cheap Web-Scale Parallel Text from the Common Crawl</i> Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch and Adam Lopez .....	1374
<i>A Sentence Compression Based Framework to Query-Focused Multi-Document Summarization</i> Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian and Claire Cardie .....	1384
<i>Domain-Independent Abstract Generation for Focused Meeting Summarization</i> Lu Wang and Claire Cardie .....	1395
<i>A Statistical NLG Framework for Aggregated Planning and Realization</i> Ravi Kondadadi, Blake Howald and Frank Schilder .....	1406
<i>Models of Translation Competitions</i> Mark Hopkins and Jonathan May .....	1416
<i>Learning a Phrase-based Translation Model from Monolingual Data with Application to Domain Adaptation</i> Jiajun Zhang and Chengqing Zong .....	1425
<i>SenseSpotting: Never let your parallel data tie you to an old domain</i> Marine Carpuat, Hal Daume III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi and Rachel Rudinger .....	1435
<i>BRAINSUP: Brainstorming Support for Creative Sentence Generation</i> Gözde Özbal, Daniele Pighin and Carlo Strapparava .....	1446

<i>Grammatical Error Correction Using Integer Linear Programming</i> Yuanbin Wu and Hwee Tou Ng .....	1456
<i>Text-Driven Toponym Resolution using Indirect Supervision</i> Michael Speriosu and Jason Baldridge .....	1466
<i>Argument Inference from Relevant Event Mentions in Chinese Argument Extraction</i> Peifeng Li, Qiaoming Zhu and Guodong Zhou .....	1477
<i>Fine-grained Semantic Typing of Emerging Entities</i> Ndapandula Nakashole, Tomasz Tylenda and Gerhard Weikum .....	1488
<i>Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction</i> Barbara Plank and Alessandro Moschitti .....	1498
<i>A joint model of word segmentation and phonological variation for English word-final /t/-deletion</i> Benjamin Börschinger, Mark Johnson and Katherine Demuth .....	1508
<i>Compositional-ly Derived Representations of Morphologically Complex Words in Distributional Semantics</i> Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli and Marco Baroni .....	1517
<i>Unsupervised Consonant-Vowel Prediction over Hundreds of Languages</i> Young-Bum Kim and Benjamin Snyder .....	1527
<i>Improving Text Simplification Language Modeling Using Unsimplified Text Data</i> David Kauchak .....	1537
<i>Combining Referring Expression Generation and Surface Realization: A Corpus-Based Investigation of Architectures</i> Sina Zarrieß and Jonas Kuhn .....	1547
<i>Named Entity Recognition using Cross-lingual Resources: Arabic as an Example</i> Kareem Darwish .....	1558
<i>Beam Search for Solving Substitution Ciphers</i> Malte Nuhn, Julian Schamper and Hermann Ney .....	1568
<i>Social Text Normalization using Contextual Graph Random Walks</i> Hany Hassan and Arul Menezes .....	1577
<i>Integrating Phrase-based Reordering Features into a Chart-based Decoder for Machine Translation</i> ThuyLinh Nguyen and Stephan Vogel .....	1587
<i>Machine Translation Detection from Monolingual Web-Text</i> Yuki Arase and Ming Zhou .....	1597
<i>Paraphrase-Driven Learning for Open Question Answering</i> Anthony Fader, Luke Zettlemoyer and Oren Etzioni .....	1608
<i>Aid is Out There: Looking for Help from Tweets during a Large Scale Disaster</i> István Varga, Motoki Sano, Kentaro Torisawa, Chikara Hashimoto, Kiyonori Ohtake, Takao Kawai, Jong-Hoon Oh and Stijn De Saeger .....	1619
<i>A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations</i> Angeliki Lazaridou, Ivan Titov and Caroline Sporleder .....	1630

<i>Joint Inference for Fine-grained Opinion Extraction</i> Bishan Yang and Claire Cardie .....	1640
<i>Linguistic Models for Analyzing and Detecting Biased Language</i> Marta Recasens, Cristian Danescu-Niculescu-Mizil and Dan Jurafsky .....	1650
<i>Evaluating a City Exploration Dialogue System with Integrated Question-Answering and Pedestrian Navigation</i> Srinivasan Janarthanam, Oliver Lemon, Phil Bartie, Tiphaine Dalmas, Anna Dickinson, Xingkun Liu, William Mackaness and Bonnie Webber .....	1660
<i>Lightly Supervised Learning of Procedural Dialog Systems</i> Svitlana Volkova, Pallavi Choudhury, Chris Quirk, Bill Dolan and Luke Zettlemoyer .....	1669
<i>Public Dialogue: Analysis of Tolerance in Online Discussions</i> Arjun Mukherjee, Vivek Venkataraman, Bing Liu and Sharon Meraz .....	1680
<i>Offspring from Reproduction Problems: What Replication Failure Teaches Us</i> Antske Fokkens, Marieke van Erp, Marten Postma, Ted Pedersen, Piek Vossen and Nuno Freire	1691
<i>Evaluating Text Segmentation using Boundary Edit Distance</i> Chris Fournier .....	1702
<i>Crowd Prefers the Middle Path: A New IAA Metric for Crowdsourcing Reveals Turker Biases in Query Segmentation</i> Rohan Ramanath, Monojit Choudhury, Kalika Bali and Rishiraj Saha Roy .....	1713
<i>Deceptive Answer Prediction with User Preference Graph</i> Fangtao Li, Yang Gao, Shuchang Zhou, Xiance Si and Decheng Dai .....	1723
<i>Why-Question Answering using Intra- and Inter-Sentential Causal Relations</i> Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger and Kiyonori Ohtake .....	1733
<i>Question Answering Using Enhanced Lexical Semantic Models</i> Wen-tau Yih, Ming-Wei Chang, Christopher Meek and Andrzej Pastusiak .....	1744
<i>Syntactic Patterns versus Word Alignment: Extracting Opinion Targets from Online Reviews</i> Kang Liu, Liheng Xu and Jun Zhao .....	1754
<i>Mining Opinion Words and Opinion Targets in a Two-Stage Framework</i> Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen and Jun Zhao .....	1764
<i>Connotation Lexicon: A Dash of Sentiment Beneath the Surface Meaning</i> Song Feng, Jun Seok Kang, Polina Kuznetsova and Yejin Choi .....	1774

# Conference Program

**Monday August 5, 2013**

**(7:30 - 17:00) Registration**

**(9:00 - 9:30) Opening session**

**(9:30) Invited Talk 1: Harald Baayen**

**(10:30) Coffee Break**

**Oral Presentations**

**(11:00 -12:15) LP 1a**

11:00 *A Shift-Reduce Parsing Algorithm for Phrase-based String-to-Dependency Translation*

Yang Liu

11:25 *Integrating Translation Memory into Phrase-Based Machine Translation during Decoding*

Kun Wang, Chengqing Zong and Keh-Yih Su

11:50 *Training Nondeficient Variants of IBM-3 and IBM-4 for Word Alignment*

Thomas Schoenemann

**(11:00 -12:15) LP 1b**

11:00 *Modelling Annotator Bias with Multi-task Gaussian Processes: An Application to Machine Translation Quality Estimation*

Trevor Cohn and Lucia Specia

11:25 *Smoothed marginal distribution constraints for language modeling*

Brian Roark, Cyril Allauzen and Michael Riley

11:50 *Grounded Language Learning from Video Described with Sentences*

Haonan Yu and Jeffrey Mark Siskind

**Monday August 5, 2013 (continued)**

**(11:00 -12:15) LP 1c**

- 11:00 *Plurality, Negation, and Quantification: Towards Comprehensive Quantifier Scope Disambiguation*  
Mehdi Manshadi, Daniel Gildea and James Allen
- 11:25 *Joint Event Extraction via Structured Prediction with Global Features*  
Qi Li, Heng Ji and Liang Huang
- 11:50 *Language-Independent Discriminative Parsing of Temporal Expressions*  
Gabor Angeli and Jakob Uszkoreit

**(11:00 -12:15) LP 1d**

- 11:00 *Graph-based Local Coherence Modeling*  
Camille Guinaudeau and Michael Strube
- 11:25 *Recognizing Rare Social Phenomena in Conversation: Empowerment Detection in Support Group Chatrooms*  
Elijah Mayfield, David Adamson and Carolyn Penstein Rosé
- 11:50 *Decentralized Entity-Level Modeling for Coreference Resolution*  
Greg Durrett, David Hall and Dan Klein

**(11:00 -12:15) LP 1e**

- 11:00 *Chinese Parsing Exploiting Characters*  
Meishan Zhang, Yue Zhang, Wanxiang Che and Ting Liu
- 11:25 *A Transition-Based Dependency Parser Using a Dynamic Parsing Strategy*  
Francesco Sartorio, Giorgio Satta and Joakim Nivre
- 11:50 *General binarization for parsing and translation*  
Matthias BÜchse, Alexander Koller and Heiko Vogler

**Monday August 5, 2013 (continued)**

**(12:15) Lunch break**

**(13:45 -15:00) LP 2a**

13:45 *Distortion Model Considering Rich Context for Statistical Machine Translation*  
Isao Goto, Masao Utiyama, Eiichiro Sumita, Akihiro Tamura and Sadao Kurohashi

14:10 *Word Alignment Modeling with Context Dependent Deep Neural Network*  
Nan Yang, Shujie Liu, Mu Li, Ming Zhou and Nenghai Yu

14:35 *Microblogs as Parallel Corpora*  
Wang Ling, Guang Xiang, Chris Dyer, Alan Black and Isabel Trancoso

**(13:45 -15:00) LP 2b**

13:45 *Improved Bayesian Logistic Supervised Topic Models with Data Augmentation*  
Jun Zhu, Xun Zheng and Bo Zhang

14:10 *Fast and Robust Compressive Summarization with Dual Decomposition and Multi-Task Learning*  
Miguel Almeida and Andre Martins

14:35 *Unsupervised Transcription of Historical Documents*  
Taylor Berg-Kirkpatrick, Greg Durrett and Dan Klein

**(13:45 -15:00) LP 2c**

13:45 *Adapting Discriminative Reranking to Grounded Language Learning*  
Joohyun Kim and Raymond Mooney

14:10 *Universal Conceptual Cognitive Annotation (UCCA)*  
Omri Abend and Ari Rappoport

14:35 *Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media*  
Weiwei Guo, Hao Li, Heng Ji and Mona Diab

**Monday August 5, 2013 (continued)**

**(13:45 -15:00) LP 2d**

- 13:45 *A computational approach to politeness with application to social factors*  
Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec and Christopher Potts
- 14:10 *Modeling Thesis Clarity in Student Essays*  
Isaac Persing and Vincent Ng
- 14:35 *Translating Italian connectives into Italian Sign Language*  
Camillo Lugaresi and Barbara Di Eugenio

**(13:45 -15:00) LP 2e**

- 13:45 *Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing*  
David Mareček and Milan Straka
- 14:10 *Transfer Learning for Constituency-Based Grammars*  
Yuan Zhang, Regina Barzilay and Amir Globerson
- 14:35 *A Context Free TAG Variant*  
Ben Swanson, Elif Yamangil, Eugene Charniak and Stuart Shieber

**(15:00 -16:15) LP 3a**

- 15:00 *Fast and Adaptive Online Training of Feature-Rich Translation Models*  
Spence Green, Sida Wang, Daniel Cer and Christopher D. Manning
- 15:25 *Advancements in Reordering Models for Statistical Machine Translation*  
Minwei Feng, Jan-Thorsten Peter and Hermann Ney
- 15:50 *A Markov Model of Machine Translation using Non-parametric Bayesian Inference*  
Yang Feng and Trevor Cohn

**Monday August 5, 2013 (continued)**

**(15:00 -16:15) LP 3b**

- 15:00 *Scaling Semi-supervised Naive Bayes with Feature Marginals*  
Michael Lucas and Doug Downey
- 15:25 *Learning Latent Personas of Film Characters*  
David Bamman, Brendan O'Connor and Noah A. Smith
- 15:50 *Scalable Decipherment for Machine Translation via Hash Sampling*  
Sujith Ravi

**(15:00 -16:15) LP 3c**

- 15:00 *Automatic Interpretation of the English Possessive*  
Stephen Tratz and Eduard Hovy
- 15:25 *Is a 204 cm Man Tall or Small ? Acquisition of Numerical Common Sense from the Web*  
Katsuma Narisawa, Yotaro Watanabe, Junta Mizuno, Naoaki Okazaki and Kentaro Inui
- 15:50 *Probabilistic Domain Modelling With Contextualized Distributional Semantic Vectors*  
Jackie Chi Kit Cheung and Gerald Penn

**(15:00 -16:15) LP 3d**

- 15:00 *Extracting bilingual terminologies from comparable corpora*  
Ahmet Aker, Monica Paramita and Rob Gaizauskas
- 15:25 *The Haves and the Have-Nots: Leveraging Unlabelled Corpora for Sentiment Analysis*  
Kashyap Popat, Balamurali A.R, Pushpak Bhattacharyya and Gholamreza Haffari
- 15:50 *Large-scale Semantic Parsing via Schema Matching and Lexicon Extension*  
Qingqing Cai and Alexander Yates

**Monday August 5, 2013 (continued)**

**(15:00 -16:15) LP 3e**

- 15:00 *Fast and Accurate Shift-Reduce Constituent Parsing*  
Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang and Jingbo Zhu
- 15:25 *Nonconvex Global Optimization for Latent-Variable Models*  
Matthew R. Gormley and Jason Eisner
- 15:50 *Parsing with Compositional Vector Grammars*  
Richard Socher, John Bauer, Christopher D. Manning and Ng Andrew Y.

**(16:15) Coffee Break**

**(18:30 - 19:45) Poster Session A**

**LP - Dialogue and Interactive Systems**

*Discriminative state tracking for spoken dialog systems*  
Angeliki Metallinou, Dan Bohus and Jason Williams

**LP- Discourse, Coreference and Pragmatics**

*Leveraging Synthetic Discourse Data via Multi-task Learning for Implicit Discourse Relation Recognition*  
Man Lan, Yu Xu and Zhengyu Niu

*Combining Intra- and Multi-sentential Rhetorical Parsing for Document-level Discourse Analysis*  
Shafiq Joty, Giuseppe Carenini, Raymond Ng and Yashar Mehdad

*Improving pairwise coreference models through feature space hierarchy learning*  
Emmanuel Lassalle and Pascal Denis

Monday August 5, 2013 (continued)

**LP - Information Retrieval**

*Feature-Based Selection of Dependency Paths in Ad Hoc Information Retrieval*

K. Tamsin Maxwell, Jon Oberlander and W. Bruce Croft

**LP - Language Resources**

*Coordination Structures in Dependency Treebanks*

Martin Popel, David Mareček, Jan Štěpánek, Daniel Zeman and Zdeněk Žabokrtský

*GlossBoot: Bootstrapping Multilingual Domain Glossaries from the Web*

Flavio De Benedictis, Stefano Faralli and Roberto Navigli

*Collective Annotation of Linguistic Resources: Basic Principles and a Formal Model*

Ulle Endriss and Raquel Fernández

*ParGramBank: The ParGram Parallel Treebank*

Sebastian Sulger, Miriam Butt, Tracy Holloway King, Paul Meurer, Tibor Laczkó, György Rákosi, Cheikh Bamba Dione, Helge Dyvik, Victoria Rosén, Koenraad De Smedt, Agnieszka Patejuk, Ozlem Cetinoglu, I Wayan Arka and Meladel Mistica

**LP - Lexical Semantics and Ontologies**

*Identifying Bad Semantic Neighbors for Improving Distributional Thesauri*

Olivier Ferret

*Models of Semantic Representation with Visual Attributes*

Carina Silberer, Vittorio Ferrari and Mirella Lapata

**Monday August 5, 2013 (continued)**

**LP - Low Resource Language Processing**

*Real-World Semi-Supervised Learning of POS-Taggers for Low-Resource Languages*

Dan Garrette, Jason Mielens and Jason Baldridge

**LP - Machine Translation: Methods, Applications and Evaluations**

*Using subcategorization knowledge to improve case prediction for translation to German*

Marion Weller, Alexander Fraser and Sabine Schulte im Walde

*Name-aware Machine Translation*

Haibo Li, Jing Zheng, Heng Ji, Qi Li and Wen Wang

*Decipherment Complexity in 1:1 Substitution Ciphers*

Malte Nuhn and Hermann Ney

*Non-Monotonic Sentence Alignment via Semisupervised Learning*

Xiaojun Quan, Chunyu Kit and Yan Song

**LP - Multilinguality**

*Bootstrapping Entity Translation on Weakly Comparable Corpora*

Taesung Lee and Seung-won Hwang

*Transfer Learning Based Cross-lingual Knowledge Extraction for Wikipedia*

Zhigang Wang, Zhixing Li, Juanzi Li, Jie Tang and Jeff Z. Pan

*Bridging Languages through Etymology: The case of cross language text categorization*

Vivi Nastase and Carlo Strapparava

**Monday August 5, 2013 (continued)**

**LP - NLP Applications**

*Creating Similarity: Lateral Thinking for Vertical Similarity Judgments*

Tony Veale and Guofu Li

*Discovering User Interactions in Ideological Discussions*

Arjun Mukherjee and Bing Liu

**LP - NLP and Creativity**

*Multilingual Affect Polarity and Valence Prediction in Metaphor-Rich Texts*

Zornitsa Kozareva

**LP - NLP for the Languages of Central and Eastern Europe and the Balkans**

*Large tagset labeling using Feed Forward Neural Networks. Case study on Romanian Language*

Tiberiu Boros, Radu Ion and Dan Tufis

*Learning to lemmatise Polish noun phrases*

Adam Radziszewski

**LP - NLP for the Web and Social Media**

*Using Conceptual Class Attributes to Characterize Social Media Users*

Shane Bergsma and Benjamin Van Durme

*The Impact of Topic Bias on Quality Flaw Prediction in Wikipedia*

Oliver Ferschke, Iryna Gurevych and Marc Rittberger

*Mining Informal Language from Chinese Microtext: Joint Word Recognition and Segmentation*

Aobo Wang and Min-Yen Kan

*Generating Synthetic Comparable Questions for News Articles*

Oleg Rokhlenko and Idan Szpektor

**Monday August 5, 2013 (continued)**

**LP - Spoken Language Processing**

*Punctuation Prediction with Transition-based Parsing*

Dongdong Zhang, Shuangzhi Wu, Nan Yang and Mu Li

**LP - Word Segmentation**

*Discriminative Learning with Natural Annotations: Word Segmentation as a Case Study*

Wenbin Jiang, Meng Sun, Yajuan Lü, Yating Yang and Qun Liu

*Graph-based Semi-Supervised Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging*

Xiaodong Zeng, Derek F. Wong, Lidia S. Chao and Isabel Trancoso

**(19:45 - 21:00) Poster Session B**

**LP - Machine Translation: Statistical Models**

*An Infinite Hierarchical Bayesian Model of Phrasal Translation*

Trevor Cohn and Gholamreza Haffari

*Additive Neural Networks for Statistical Machine Translation*

lemao liu, Taro Watanabe, Eiichiro Sumita and Tiejun Zhao

*Hierarchical Phrase Table Combination for Machine Translation*

Conghui Zhu, Taro Watanabe, Eiichiro Sumita and Tiejun Zhao

*Shallow Local Multi-Bottom-up Tree Transducers in Statistical Machine Translation*

Fabienne Braune, Nina Seemann, Daniel Quernheim and Andreas Maletti

*Enlisting the Ghost: Modeling Empty Categories for Machine Translation*

Bing Xiang, Xiaoqiang Luo and Bowen Zhou

*A Multi-Domain Translation Model Framework for Statistical Machine Translation*

Rico Sennrich, Holger Schwenk and Walid Aransa

*Part-of-Speech Induction in Dependency Trees for Statistical Machine Translation*

Akihiro Tamura, Taro Watanabe, Eiichiro Sumita, Hiroya Takamura and Manabu Okumura

**Monday August 5, 2013 (continued)**

**LP - Question Answering**

*Statistical Machine Translation Improves Question Retrieval in Community Question Answering via Matrix Factorization*

Guangyou Zhou, Fang Liu, Yang Liu, Shizhu He and Jun Zhao

**LP - Semantics**

*Improved Lexical Acquisition through DPP-based Verb Clustering*

Roi Reichart and Anna Korhonen

*Semantic Frames to Predict Stock Price Movement*

Boyi Xie, Rebecca J. Passonneau, Leon Wu and Germán G. Creamer

*Density Maximization in Context-Sense Metric Space for All-words WSD*

Koichi Tanigaki, Mitsuteru Shiba, Tatsuji Munaka and Yoshinori Sagisaka

*The Role of Syntax in Vector Space Models of Compositional Semantics*

Karl Moritz Hermann and Phil Blunsom

*Margin-based Decomposed Amortized Inference*

Gourab Kundu, Vivek Srikumar and Dan Roth

*Semi-Supervised Semantic Tagging of Conversational Understanding using Markov Topic Regression*

Asli Celikyilmaz, Dilek Hakkani-Tur, Gokhan Tur and Ruhi Sarikaya

*Parsing Graphs with Hyperedge Replacement Grammars*

David Chiang, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, Bevan Jones and Kevin Knight

*Grounded Unsupervised Semantic Parsing*

Hoifung Poon

**Monday August 5, 2013 (continued)**

**LP - Sentiment Analysis, Opinion Mining and Text Classification**

*Automatic detection of deception in child-produced speech using syntactic complexity features*

Maria Yancheva and Frank Rudzicz

*Sentiment Relevance*

Christian Scheible and Hinrich Schütze

*Predicting and Eliciting Addressee's Emotion in Online Dialogue*

Takayuki Hasegawa, Nobuhiro Kaji, Naoki Yoshinaga and Masashi Toyoda

*Utterance-Level Multimodal Sentiment Analysis*

Veronica Perez-Rosas, Rada Mihalcea and Louis-Philippe Morency

*Probabilistic Sense Sentiment Similarity through Hidden Emotions*

Mitra Mohtarami, Man Lan and Chew Lim Tan

**LP - Statistical and Machine Learning Methods in NLP**

*A user-centric model of voting intention from Social Media*

Vasileios Lampos, Daniel Preoțiuc-Pietro and Trevor Cohn

**LP - Summarization and Generation**

*Using Supervised Bigram-based ILP for Extractive Summarization*

Chen Li, Xian Qian and Yang Liu

*Summarization Through Submodularity and Dispersion*

Anirban Dasgupta, Ravi Kumar and Sujith Ravi

*Subtree Extractive Summarization via Submodular Maximization*

Hajime Morita, Ryohei Sasano, Hiroya Takamura and Manabu Okumura

**Monday August 5, 2013 (continued)**

**LP - Syntax and Parsing**

*The effect of non-tightness on Bayesian estimation of PCFGs*

Shay B. Cohen and Mark Johnson

*Integrating Multiple Dependency Corpora for Inducing Wide-coverage Japanese CCG Resources*

Sumire Uematsu, Takuya Matsuzaki, Hiroki Hanaoka, Yusuke Miyao and Hideki Mima

*Transition-based Dependency Parsing with Selectional Branching*

Jinho D. Choi and Andrew McCallum

*Bilingually-Guided Monolingual Dependency Grammar Induction*

Kai Liu, Yajuan Lü, Wenbin Jiang and Qun Liu

**LP - Tagging and Chunking**

*Joint Word Alignment and Bilingual Named Entity Recognition Using Dual Decomposition*

Mengqiu Wang, Wanxiang Che and Christopher D. Manning

*Resolving Entity Morphs in Censored Data*

Hongzhao Huang, Zhen Wen, Dian Yu, Heng Ji, Yizhou Sun, Jiawei Han and He Li

**LP - Text Mining and Information Extraction**

*Learning to Extract International Relations from Political Context*

Brendan O'Connor, Brandon M. Stewart and Noah A. Smith

**Tuesday August 6, 2013**

**(7:30 - 17:00) Registration**

**(9:00) Industrial Lecture: Lars Rasmussen (Facebook)**

**(10:00) Best Paper Award**

**(10:30) Coffee Break**

**Oral Presentations**

**(11:00 -12:15) LP 5a**

11:00 *Graph Propagation for Paraphrasing Out-of-Vocabulary Words in Statistical Machine Translation*

Majid Razmara, Maryam Siahbani, Reza Haffari and Anoop Sarkar

11:25 *Online Relative Margin Maximization for Statistical Machine Translation*

Vladimir Eidelman, Yuval Marton and Philip Resnik

11:50 *Handling Ambiguities of Bilingual Predicate-Argument Structures for Statistical Machine Translation*

Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong

**(11:00 -12:15) LP 5b**

11:00 *Reconstructing an Indo-European Family Tree from Non-native English Texts*

Ryo Nagata and Edward Whittaker

11:25 *Word Association Profiles and their Use for Automated Scoring of Essays*

Beata Beigman Klebanov and Michael Flor

11:50 *Adaptive Parser-Centric Text Normalization*

Congle Zhang, Tyler Baldwin, Howard Ho, Benny Kimelfeld and Yunyao Li

**Tuesday August 6, 2013 (continued)**

**(11:00 -12:15) LP 5c**

- 11:00 *A Random Walk Approach to Selectional Preferences Based on Preference Ranking and Propagation*  
Zhenhua Tian, Hengheng Xiang, Ziqi Liu and Qinghua Zheng
- 11:25 *ImpAr: A Deterministic Algorithm for Implicit Semantic Role Labelling*  
Egoitz Laparra and German Rigau
- 11:50 *Cross-lingual Transfer of Semantic Role Labeling Models*  
Mikhail Kozhevnikov and Ivan Titov

**(11:00 -12:15) LP 5d**

- 11:00 *DERivBase: Inducing and Evaluating a Derivational Morphology Resource for German*  
Britta Zeller, Jan Šnajder and Sebastian Padó
- 11:25 *Crowdsourcing Interaction Logs to Understand Text Reuse from the Web*  
Martin Potthast, Matthias Hagen, Michael Völske and Benno Stein
- 11:50 *SPred: Large-scale Harvesting of Semantic Predicates*  
Tiziano Flati and Roberto Navigli

**(11:00 -12:15) LP 5e**

- 11:00 *Towards Robust Abstractive Multi-Document Summarization: A Caseframe Analysis of Centrality and Domain*  
Jackie Chi Kit Cheung and Gerald Penn
- 11:25 *HEADY: News headline abstraction through event pattern clustering*  
Enrique Alfonseca, Daniele Pighin and Guillermo Garrido
- 11:50 *Conditional Random Fields for Responsive Surface Realisation using Global Features*  
Nina Dethlefs, Helen Hastie, Heriberto Cuayáhuitl and Oliver Lemon

**Tuesday August 6, 2013 (continued)**

**(12:15) Lunch break**

**(13:45 -15:00) LP 6a**

13:45 *Two-Neighbor Orientation Model with Cross-Boundary Global Contexts*  
Hendra Setiawan, Bowen Zhou, Bing Xiang and Libin Shen

14:10 *Cut the noise: Mutually reinforcing reordering and alignments for improved machine translation*  
Karthik Visweswariah, Mitesh M. Khapra and Ananthakrishnan Ramanathan

14:35 *Vector Space Model for Adaptation in Statistical Machine Translation*  
Boxing Chen, Roland Kuhn and George Foster

**(13:45 -15:00) LP 6b**

13:45 *From Natural Language Specifications to Program Input Parsers*  
Tao Lei, Fan Long, Regina Barzilay and Martin Rinard

14:10 *Entity Linking for Tweets*  
Xiaohua Liu, Yitong Li, Haocheng Wu, Ming Zhou, Furu Wei and Yi Lu

14:35 *Identification of Speakers in Novels*  
Hua He, Denilson Barbosa and Grzegorz Kondrak

**(13:45 -15:00) LP 6c**

13:45 *Language Acquisition and Probabilistic Models: keeping it simple*  
Aline Villavicencio, Marco Idiart, Robert Berwick and Igor Malioutov

14:10 *A Two Level Model for Context Sensitive Inference Rules*  
Oren Melamud, Jonathan Berant, Ido Dagan, Jacob Goldberger and Idan Szpektor

14:35 *Align, Disambiguate and Walk: A Unified Approach for Measuring Semantic Similarity*  
Mohammad Taher Pilehvar, David Jurgens and Roberto Navigli

**Tuesday August 6, 2013 (continued)**

**(13:45 -15:00) LP 6d**

- 13:45 *Linking and Extending an Open Multilingual Wordnet*  
Francis Bond and Ryan Foster
- 14:10 *FrameNet on the Way to Babel: Creating a Bilingual FrameNet Using Wiktionary as Interlingual Connection*  
Silvana Hartmann and Iryna Gurevych
- 14:35 *Dirt Cheap Web-Scale Parallel Text from the Common Crawl*  
Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch and Adam Lopez

**(13:45 -15:00) LP 6e**

- 13:45 *A Sentence Compression Based Framework to Query-Focused Multi-Document Summarization*  
Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian and Claire Cardie
- 14:10 *Domain-Independent Abstract Generation for Focused Meeting Summarization*  
Lu Wang and Claire Cardie
- 14:35 *A Statistical NLG Framework for Aggregated Planning and Realization*  
Ravi Kondadadi, Blake Howald and Frank Schilder

**(15:00 -16:15) LP 7a**

- 15:00 *Models of Translation Competitions*  
Mark Hopkins and Jonathan May
- 15:25 *Learning a Phrase-based Translation Model from Monolingual Data with Application to Domain Adaptation*  
Jiajun Zhang and Chengqing Zong
- 15:50 *SenseSpotting: Never let your parallel data tie you to an old domain*  
Marine Carpuat, Hal Daume III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi and Rachel Rudinger

**Tuesday August 6, 2013 (continued)**

**(15:00 -16:15) LP 7b**

15:00 *BRAINSUP: Brainstorming Support for Creative Sentence Generation*  
Gözde Özbal, Daniele Pighin and Carlo Strapparava

15:25 *Grammatical Error Correction Using Integer Linear Programming*  
Yuanbin Wu and Hwee Tou Ng

15:50 *Text-Driven Toponym Resolution using Indirect Supervision*  
Michael Speriosu and Jason Baldridge

**(15:00 -16:15) LP 7c**

15:00 *Argument Inference from Relevant Event Mentions in Chinese Argument Extraction*  
Peifeng Li, Qiaoming Zhu and Guodong Zhou

15:25 *Fine-grained Semantic Typing of Emerging Entities*  
Ndapandula Nakashole, Tomasz Tylanda and Gerhard Weikum

15:50 *Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction*  
Barbara Plank and Alessandro Moschitti

**(15:00 -16:15) LP 7d**

15:00 *A joint model of word segmentation and phonological variation for English word-final /t/-deletion*  
Benjamin Börschinger, Mark Johnson and Katherine Demuth

15:25 *Compositional-ly Derived Representations of Morphologically Complex Words in Distributional Semantics*  
Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli and Marco Baroni

15:50 *Unsupervised Consonant-Vowel Prediction over Hundreds of Languages*  
Young-Bum Kim and Benjamin Snyder

**Tuesday August 6, 2013 (continued)**

**(15:00 -16:15) LP 7e**

15:00 *Improving Text Simplification Language Modeling Using Unsimplified Text Data*  
David Kauchak

15:25 *Combining Referring Expression Generation and Surface Realization: A Corpus-Based Investigation of Architectures*  
Sina Zarrieß and Jonas Kuhn

15:50 *Named Entity Recognition using Cross-lingual Resources: Arabic as an Example*  
Kareem Darwish

**(16:15) Coffee Break**

**(18:30) Banquet**

**Wednesday August 7, 2013**

**(9:30) Invited Talk 3: Chantal Prat**

**(10:30) Coffee Break**

**Oral Presentations**

**(11:00 -12:15) LP 9a**

11:00 *Beam Search for Solving Substitution Ciphers*  
Malte Nuhn, Julian Schamper and Hermann Ney

11:25 *Social Text Normalization using Contextual Graph Random Walks*  
Hany Hassan and Arul Menezes

11:50 *Integrating Phrase-based Reordering Features into a Chart-based Decoder for Machine Translation*  
ThuyLinh Nguyen and Stephan Vogel

**Wednesday August 7, 2013 (continued)**

**(11:00 -12:15) LP 9b**

- 11:00 *Machine Translation Detection from Monolingual Web-Text*  
Yuki Arase and Ming Zhou
- 11:25 *Paraphrase-Driven Learning for Open Question Answering*  
Anthony Fader, Luke Zettlemoyer and Oren Etzioni
- 11:50 *Aid is Out There: Looking for Help from Tweets during a Large Scale Disaster*  
István Varga, Motoki Sano, Kentaro Torisawa, Chikara Hashimoto, Kiyonori Ohtake,  
Takao Kawai, Jong-Hoon Oh and Stijn De Saeger

**(11:00 -12:15) LP 9c**

- 11:00 *A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations*  
Angeliki Lazaridou, Ivan Titov and Caroline Sporleder
- 11:25 *Joint Inference for Fine-grained Opinion Extraction*  
Bishan Yang and Claire Cardie
- 11:50 *Linguistic Models for Analyzing and Detecting Biased Language*  
Marta Recasens, Cristian Danescu-Niculescu-Mizil and Dan Jurafsky

**(11:00 -12:15) LP 9d**

- 11:00 *Evaluating a City Exploration Dialogue System with Integrated Question-Answering and Pedestrian Navigation*  
Srinivasan Janarthanam, Oliver Lemon, Phil Bartie, Tiphaine Dalmas, Anna Dickinson,  
Xingkun Liu, William Mackaness and Bonnie Webber
- 11:25 *Lightly Supervised Learning of Procedural Dialog Systems*  
Svitlana Volkova, Pallavi Choudhury, Chris Quirk, Bill Dolan and Luke Zettlemoyer
- 11:50 *Public Dialogue: Analysis of Tolerance in Online Discussions*  
Arjun Mukherjee, Vivek Venkataraman, Bing Liu and Sharon Meraz

**Wednesday August 7, 2013 (continued)**

**(12:15) Lunch break**

**(13:30) ACL Business Meeting**

**(15:00 -16:15) LP 10a**

15:00 *Offspring from Reproduction Problems: What Replication Failure Teaches Us*  
Antske Fokkens, Marieke van Erp, Marten Postma, Ted Pedersen, Piek Vossen and Nuno Freire

15:25 *Evaluating Text Segmentation using Boundary Edit Distance*  
Chris Fournier

15:50 *Crowd Prefers the Middle Path: A New IAA Metric for Crowdsourcing Reveals Turker Biases in Query Segmentation*  
Rohan Ramanath, Monojit Choudhury, Kalika Bali and Rishiraj Saha Roy

**(15:00 -16:15) LP 10b**

15:00 *Deceptive Answer Prediction with User Preference Graph*  
Fangtao Li, Yang Gao, Shuchang Zhou, Xiance Si and Decheng Dai

15:25 *Why-Question Answering using Intra- and Inter-Sentential Causal Relations*  
Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger and Kiyonori Ohtake

15:50 *Question Answering Using Enhanced Lexical Semantic Models*  
Wen-tau Yih, Ming-Wei Chang, Christopher Meek and Andrzej Pastusiak

**Wednesday August 7, 2013 (continued)**

**(15:00 -16:15) LP 10c**

15:00 *Syntactic Patterns versus Word Alignment: Extracting Opinion Targets from Online Reviews*

Kang Liu, Liheng Xu and Jun Zhao

15:25 *Mining Opinion Words and Opinion Targets in a Two-Stage Framework*

Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen and Jun Zhao

15:50 *Connotation Lexicon: A Dash of Sentiment Beneath the Surface Meaning*

Song Feng, Jun Seok Kang, Polina Kuznetsova and Yejin Choi

**(16:15) Coffee Break**

**(18:30) Lifetime Achievement Award Session**

**(19:15) Closing Session**

**(19:30) End**

# A Shift-Reduce Parsing Algorithm for Phrase-based String-to-Dependency Translation

Yang Liu

State Key Laboratory of Intelligent Technology and Systems  
Tsinghua National Laboratory for Information Science and Technology  
Department of Computer Science and Technology  
Tsinghua University, Beijing 100084, China  
liuyang2011@tsinghua.edu.cn

## Abstract

We introduce a shift-reduce parsing algorithm for phrase-based string-to-dependency translation. As the algorithm generates dependency trees for partial translations left-to-right in decoding, it allows for efficient integration of both  $n$ -gram and dependency language models. To resolve conflicts in shift-reduce parsing, we propose a maximum entropy model trained on the derivation graph of training data. As our approach combines the merits of phrase-based and string-to-dependency models, it achieves significant improvements over the two baselines on the NIST Chinese-English datasets.

## 1 Introduction

Modern statistical machine translation approaches can be roughly divided into two broad categories: *phrase-based* and *syntax-based*. Phrase-based approaches treat phrase, which is usually a sequence of consecutive words, as the basic unit of translation (Koehn et al., 2003; Och and Ney, 2004). As phrases are capable of memorizing local context, phrase-based approaches excel at handling local word selection and reordering. In addition, it is straightforward to integrate  $n$ -gram language models into phrase-based decoders in which translation always grows left-to-right. As a result, phrase-based decoders only need to maintain the boundary words on one end to calculate language model probabilities. However, as phrase-based decoding usually casts translation as a string concatenation problem and permits arbitrary permutation, it proves to be NP-complete (Knight, 1999).

Syntax-based approaches, on the other hand, model the hierarchical structure of natural languages (Wu, 1997; Yamada and Knight, 2001; Chiang, 2005; Quirk et al., 2005; Galley et al.,

2006; Liu et al., 2006; Huang et al., 2006; Shen et al., 2008; Mi and Huang, 2008; Zhang et al., 2008). As syntactic information can be exploited to provide linguistically-motivated reordering rules, predicting non-local permutation is computationally tractable in syntax-based approaches. Unfortunately, as syntax-based decoders often generate target-language words in a bottom-up way using the CKY algorithm, integrating  $n$ -gram language models becomes more expensive because they have to maintain target boundary words at both ends of a partial translation (Chiang, 2007; Huang and Chiang, 2007). Moreover, syntax-based approaches often suffer from the rule coverage problem since syntactic constraints rule out a large portion of non-syntactic phrase pairs, which might help decoders generalize well to unseen data (Marcu et al., 2006). Furthermore, the introduction of non-terminals makes the grammar size significantly bigger than phrase tables and leads to higher memory requirement (Chiang, 2007).

As a result, incremental decoding with hierarchical structures has attracted increasing attention in recent years. While some authors try to integrate syntax into phrase-based decoding (Galley and Manning, 2008; Galley and Manning, 2009; Feng et al., 2010), others develop incremental algorithms for syntax-based models (Watanabe et al., 2006; Huang and Mi, 2010; Dyer and Resnik, 2010; Feng et al., 2012). Despite these successful efforts, challenges still remain for both directions. While parsing algorithms can be used to parse partial translations in phrase-based decoding, the search space is significantly enlarged since there are exponentially many parse trees for exponentially many translations. On the other hand, although target words can be generated left-to-right by altering the way of tree transversal in syntax-based models, it is still difficult to reach full rule coverage as compared with phrase table.

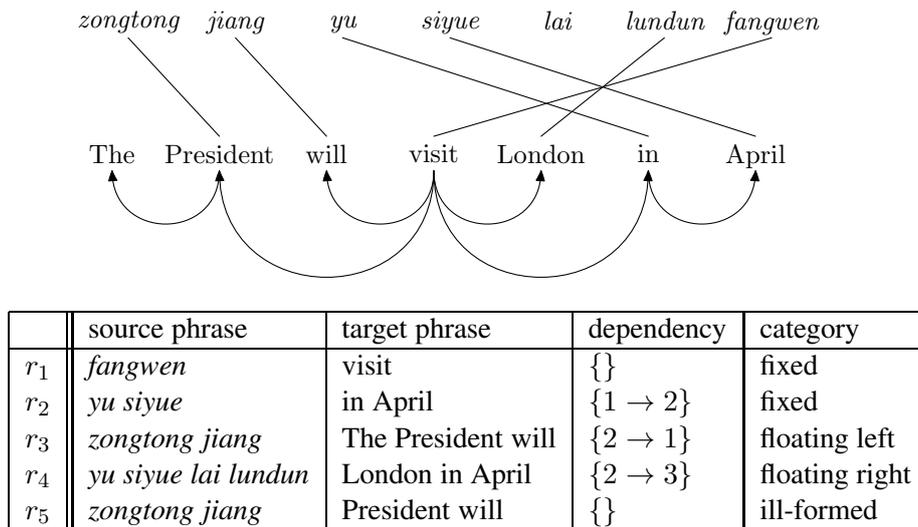


Figure 1: A training example consisting of a (romanized) Chinese sentence, an English dependency tree, and the word alignment between them. Each translation rule is composed of a source phrase, a target phrase with a set of dependency arcs. Following Shen et al. (2008), we distinguish between *fixed*, *floating*, and *ill-formed* structures.

In this paper, we propose a shift-reduce parsing algorithm for phrase-based string-to-dependency translation. The basic unit of translation in our model is *string-to-dependency phrase pair*, which consists of a phrase on the source side and a dependency structure on the target side. The algorithm generates well-formed dependency structures for partial translations left-to-right using string-to-dependency phrase pairs. Therefore, our approach is capable of combining the advantages of both phrase-based and syntax-based approaches:

1. **compact rule table:** our rule table is a subset of the original string-to-dependency grammar (Shen et al., 2008; Shen et al., 2010) by excluding rules with non-terminals.
2. **full rule coverage:** all phrase pairs, both syntactic and non-syntactic, can be used in our algorithm. This is the same with Moses (Koehn et al., 2007).
3. **efficient integration of  $n$ -gram language model:** as translation grows left-to-right in our algorithm, integrating  $n$ -gram language models is straightforward.
4. **exploiting syntactic information:** as the shift-reduce parsing algorithm generates target language dependency trees in decoding, dependency language models (Shen et al., 2008; Shen et al., 2010) can be used to encourage linguistically-motivated reordering.

5. **resolving local parsing ambiguity:** as dependency trees for phrases are memorized in rules, our approach avoids resolving local parsing ambiguity and explores in a smaller search space than parsing word-by-word on the fly in decoding (Galley and Manning, 2009).

We evaluate our method on the NIST Chinese-English translation datasets. Experiments show that our approach significantly outperforms both phrase-based (Koehn et al., 2007) and string-to-dependency approaches (Shen et al., 2008) in terms of BLEU and TER.

## 2 Shift-Reduce Parsing for Phrase-based String-to-Dependency Translation

Figure 1 shows a training example consisting of a (romanized) Chinese sentence, an English dependency tree, and the word alignment between them. Following Shen et al. (2008), string-to-dependency rules without non-terminals can be extracted from the training example. As shown in Figure 1, each rule is composed of a source phrase and a target dependency structure. Shen et al. (2008) divide dependency structures into two broad categories:

### 1. well-formed

- (a) **fixed:** the head is known or fixed;

step	action	rule	stack	coverage
0				○ ○ ○ ○ ○ ○ ○ ○
1	$S$	$r_3$	[The President will]	● ● ○ ○ ○ ○ ○ ○
2	$S$	$r_1$	[The President will] [visit]	● ● ○ ○ ○ ○ ●
3	$R_l$		[The President will visit]	● ● ○ ○ ○ ○ ●
4	$S$	$r_4$	[The President will visit] [London in April]	● ● ● ● ● ● ●
5	$R_r$		[The President will visit London in April]	● ● ● ● ● ● ●

Figure 2: Shift-reduce parsing with string-to-dependency phrase pairs. For each state, the algorithm maintains a stack to store items (i.e., well-formed dependency structures). At each step, it chooses one action to extend a state: shift ( $S$ ), reduce left ( $R_l$ ), or reduce right ( $R_r$ ). The decoding process terminates when all source words are covered and there is a complete dependency tree in the stack.

(b) **floating**: sibling nodes of a common head, but the head itself is unspecified or floating. Each of the siblings must be a complete constituent.

2. **ill-formed**: neither fixed nor floating.

We further distinguish between *left* and *right* floating structures according to the position of head. For example, as “The President will” is the left dependant of its head “visit”, it is a left floating structure.

To integrate the advantages of phrase-based and string-to-dependency models, we propose a shift-reduce algorithm for phrase-based string-to-dependency translation.

Figure 2 shows an example. We describe a *state* (i.e., parser configuration) as a tuple  $\langle \mathcal{S}, \mathcal{C} \rangle$  where  $\mathcal{S}$  is a *stack* that stores *items* and  $\mathcal{C}$  is a *coverage vector* that indicates which source words have been translated. Each item  $s \in \mathcal{S}$  is a well-formed dependency structure. The algorithm starts with an empty state. At each step, it chooses one of the three actions (Huang et al., 2009) to extend a state:

1. **shift** ( $S$ ): move a target dependency structure onto the stack;

2. **reduce left** ( $R_l$ ): combine the two items on the stack,  $s_t$  and  $s_{t-1}$  ( $t \geq 2$ ), with the root of  $s_t$  as the head and replace them with a combined item;

3. **reduce right** ( $R_r$ ): combine the two items on the stack,  $s_t$  and  $s_{t-1}$  ( $t \geq 2$ ), with the root of  $s_{t-1}$  as the head and replace them with a combined item.

The decoding process terminates when all source words are covered and there is a complete dependency tree in the stack.

Note that unlike monolingual shift-reduce parsers (Nivre, 2004; Zhang and Clark, 2008; Huang et al., 2009), our algorithm does not maintain a queue for remaining words of the input because the future dependency structure to be shifted is unknown in advance in the translation scenario. Instead, we use a coverage vector on the source side to determine when to terminate the algorithm.

For an input sentence of  $J$  words, the number of actions is  $2K - 1$ , where  $K$  is the number of rules used in decoding.<sup>1</sup> There are always  $K$  shifts and

<sup>1</sup>Empirically, we find that the average number of stacks for  $J$  words is about  $1.5 \times J$  on the Chinese-English data.

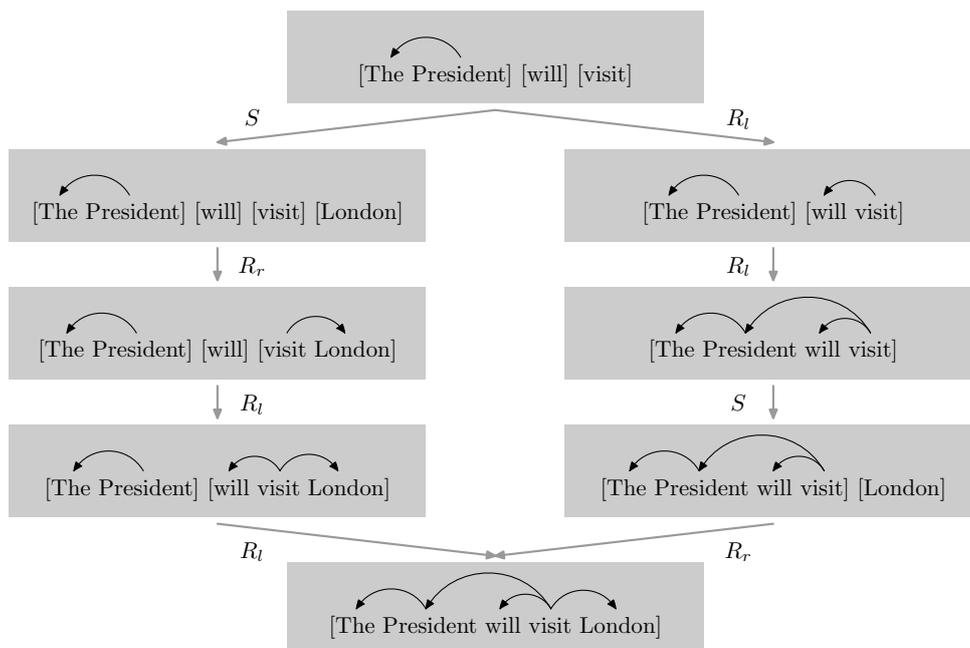


Figure 3: Ambiguity in shift-reduce parsing.

$s_{t-1}$	$s_t$	legal	action(s)
		yes	$S$
	h	yes	$S$
	l	yes	$S$
	r	no	
h	h	yes	$S, R_l, R_r$
h	l	yes	$S$
h	r	yes	$R_r$
l	h	yes	$R_l$
l	l	yes	$S$
l	r	no	
r	h	no	
r	l	no	
r	r	no	

Table 1: Conflicts in shift-reduce parsing.  $s_t$  and  $s_{t-1}$  are the top two items in the stack of a state. We use “h” to denote fixed structure, “l” to denote left floating structure, and “r” to denote right floating structure. It is clear that only “h+h” is ambiguous.

$K - 1$  reductions.

It is easy to verify that the reduce left and reduce right actions are equivalent to the left adjoining and right adjoining operations defined by Shen et al. (2008). They suffice to operate on well-formed structures and produce projective dependency parse trees.

Therefore, with dependency structures present in the stacks, it is possible to use dependency language models to encourage linguistically plausible phrase reordering.

### 3 A Maximum Entropy Based Shift-Reduce Parsing Model

Shift-reduce parsing is efficient but suffers from parsing errors caused by syntactic ambiguity. Figure 3 shows two (partial) derivations for a dependency tree. Consider the item on the top, the algorithm can either apply a shift action to move a new item or apply a reduce left action to obtain a bigger structure. This is often referred to as **conflict** in the shift-reduce dependency parsing literature (Huang et al., 2009). In this work, the shift-reduce parser faces four types of conflicts:

1. shift vs. shift;
2. shift vs. reduce left;
3. shift vs. reduce right;
4. reduce left vs. reduce right.

Fortunately, if we distinguish between left and right floating structures, it is possible to rule out most conflicts. Table 1 shows the relationship between conflicts, dependency structures and actions. We use  $s_t$  and  $s_{t-1}$  to denote the top two

[The President will visit London][in April]
   
 DT NNP MD VB NNP IN IN

type	feature templates		
Unigram	$c$	$W_h(s_t)$	$W_h(s_{t-1})$
	$W_{lc}(s_t)$	$W_{rc}(s_{t-1})$	$T_h(s_t)$
	$T_h(s_{t-1})$	$T_{lc}(s_t)$	$T_{rc}(s_{t-1})$
Bigram	$W_h(s_t) \circ W_h(s_{t-1})$	$T_h(s_t) \circ T_h(s_{t-1})$	$W_h(s_t) \circ T_h(s_t)$
	$W_h(s_{t-1}) \circ T_h(s_{t-1})$	$W_h(s_t) \circ W_{rc}(s_{t-1})$	$W_h(s_{t-1}) \circ W_{lc}(s_t)$
Trigram	$c \circ W_h(s_t) \circ W_h(s_{t-1})$	$c \circ T_h(s_t) \circ T_h(s_{t-1})$	$W_h(s_t) \circ W_h(s_{t-1}) \circ T_{lc}(s_t)$
	$W_h(s_t) \circ W_h(s_{t-1}) \circ T_{rc}(s_{t-1})$	$T_h(s_t) \circ T_h(s_{t-1}) \circ T_{lc}(s_t)$	$T_h(s_t) \circ T_h(s_{t-1}) \circ T_{rc}(s_{t-1})$

Figure 4: Feature templates for maximum entropy based shift-reduce parsing model.  $c$  is a boolean value that indicate whether all source words are covered (shift is prohibited if true),  $W_h(\cdot)$  and  $T_h(\cdot)$  are functions that get the root word and tag of an item,  $W_{lc}(\cdot)$  and  $T_{lc}(\cdot)$  returns the word and tag of the left most child of the root,  $W_{rc}(\cdot)$  and  $T_{rc}(\cdot)$  returns the word and tag of the right most child of the root. Symbol  $\circ$  denotes feature conjunction. In this example,  $c = true$ ,  $W_h(s_t) = in$ ,  $T_h(s_t) = IN$ ,  $W_h(s_{t-1}) = visit$ ,  $W_{lc}(s_{t-1}) = London$ .

items in the stack. “h” stands for fixed structure, “l” for left floating structure, and “r” for right floating structure. If the stack is empty, the only applicable action is shift. If there is only one item in the stack and the item is either fixed or left floating, the only applicable action is shift. Note that it is illegal to shift a right floating structure onto an empty stack because it will never be reduced. If the stack contains at least two items, only “h+h” is ambiguous and the others are either unambiguous or illegal. Therefore, we only need to focus on how to resolve conflicts for the “h+h” case (i.e., the top two items in a stack are both fixed structures).

We propose a maximum entropy model to resolve the conflicts for “h+h”:<sup>2</sup>

$$P_\theta(a|c, s_t, s_{t-1}) = \frac{\exp(\theta \cdot h(a, c, s_t, s_{t-1}))}{\sum_a \exp(\theta \cdot h(a, c, s_t, s_{t-1}))}$$

where  $a \in \{S, R_l, R_r\}$  is an action,  $c$  is a boolean value that indicates whether all source words are covered (shift is prohibited if true),  $s_t$  and  $s_{t-1}$  are the top two items on the stack,  $h(a, c, s_t, s_{t-1})$  is a vector of binary features and  $\theta$  is a vector of feature weights.

Figure 4 shows the feature templates used in our experiments.  $W_h(\cdot)$  and  $T_h(\cdot)$  are functions that get the root word and tag of an item,  $W_{lc}(\cdot)$  and  $T_{lc}(\cdot)$  returns the word and tag of the left most child of the root,  $W_{rc}(\cdot)$  and  $T_{rc}(\cdot)$  returns the

<sup>2</sup>The shift-shift conflicts always exist because there are usually multiple rules that can be shifted. This can be resolved using standard features in phrase-based models.

word and tag of the right most child of the root. In this example,  $c = true$ ,  $W_h(s_t) = in$ ,  $T_h(s_t) = IN$ ,  $W_h(s_{t-1}) = visit$ ,  $W_{lc}(s_{t-1}) = London$ .

To train the model, we need an “oracle” or gold-standard action sequence for each training example. Unfortunately, such oracle turns out to be non-unique even for monolingual shift-reduce dependency parsing (Huang et al., 2009). The situation for phrase-based shift-reduce parsing aggravates because there are usually multiple ways of segmenting sentence into phrases.

To alleviate this problem, we introduce a structure called **derivation graph** to compactly represent all derivations of a training example. Figure 3 shows a (partial) derivation graph, in which a node corresponds to a state and an edge corresponds to an action. The graph begins with an empty state and ends with the given training example.

More formally, a derivation graph is a directed acyclic graph  $G = \langle V, E \rangle$  where  $V$  is a set of nodes and  $E$  is a set of edges. Each node  $v$  corresponds to a state in the shift-reduce parsing process. There are two distinguished nodes:  $v_0$ , the starting empty state, and  $v_{|V|}$ , the ending completed state. Each edge  $e = (a, i, j)$  transits node  $v_i$  to node  $v_j$  via an action  $a \in \{S, R_l, R_r\}$ .

To build the derivation graph, our algorithm starts with an empty state and iteratively extends an unprocessed state until reaches the completed state. During the process, states that violate the training example are discarded. Even so, there are still exponentially many states for a training example, especially for long sentences. Fortunately, we

---

**Algorithm 1** Beam-search shift-reduce parsing.

---

```
1: procedure PARSE( $\mathbf{f}$ )
2:    $\mathcal{V} \leftarrow \emptyset$ 
3:   ADD( $v_0, \mathcal{V}[0]$ )
4:    $k \leftarrow 0$ 
5:   while  $\mathcal{V}[k] \neq \emptyset$  do
6:     for all  $v \in \mathcal{V}[k]$  do
7:       for all  $a \in \{S, R_l, R_r\}$  do
8:         EXTEND( $\mathbf{f}, v, a, \mathcal{V}$ )
9:       end for
10:    end for
11:     $k \leftarrow k + 1$ 
12:  end while
13: end procedure
```

---

only need to focus on “h+h” states. In addition, we follow Huang et al. (2009) to use the heuristic of “shortest stack” to always prefer  $R_l$  to  $S$ .

## 4 Decoding

Our decoder is based on a linear model (Och, 2003) with the following features:

1. relative frequencies in two directions;
2. lexical weights in two directions;
3. phrase penalty;
4. distance-based reordering model;
5. lexicaized reordering model;
6.  $n$ -gram language model model;
7. word penalty;
8. ill-formed structure penalty;
9. dependency language model;
10. maximum entropy parsing model.

In practice, we extend deterministic shift-reduce parsing with beam search (Zhang and Clark, 2008; Huang et al., 2009). As shown in Algorithm 1, the algorithm maintains a list of stacks  $\mathcal{V}$  and each stack groups states with the same number of accumulated actions (line 2). The stack list  $\mathcal{V}$  initializes with an empty state  $v_0$  (line 3). Then, the states in the stack are iteratively extended until there are no incomplete states (lines 4-12). The search space is constrained by discarding any state that has a score worse than:

1.  $\beta$  multiplied with the best score in the stack, or
2. the score of  $b$ -th best state in the stack.

As the stack of a state keeps changing during the decoding process, the context information needed to calculate dependency language model and maximum entropy model probabilities (e.g., root word, leftmost child, etc.) changes dynamically as well. As a result, the chance of risk-free *hypothesis recombination* (Koehn et al., 2003) significantly decreases because complicated contextual information is much less likely to be identical.

Therefore, we use *hypergraph reranking* (Huang and Chiang, 2007; Huang, 2008), which proves to be effective for integrating non-local features into dynamic programming, to alleviate this problem. The decoding process is divided into two passes. In the first pass, only **standard features** (i.e., features 1-7 in the list in the beginning of this section) are used to produce a hypergraph.<sup>3</sup> In the second pass, we use the hypergraph reranking algorithm (Huang, 2008) to find promising translations using additional **dependency features** (i.e., features 8-10 in the list). As hypergraph is capable of storing exponentially many derivations compactly, the negative effect of propagating mistakes made in the first pass to the second pass can be minimized.

To improve rule coverage, we follow Shen et al. (2008) to use ill-formed structures in decoding. If an ill-formed structure has a single root, it can be treated as a (pseudo) fixed structure; otherwise it is transformed to one (pseudo) left floating structure and one (pseudo) right floating structure. We use a feature to count how many ill-formed structures are used in decoding.

## 5 Experiments

We evaluated our phrase-based string-to-dependency translation system on Chinese-English translation. The training data consists of 2.9M pairs of sentences with 76.0M Chinese words and 82.2M English words. We used the Stanford parser (Klein and Manning, 2003) to get dependency trees for English sentences. We used the SRILM toolkit (Stolcke, 2002) to train a

---

<sup>3</sup>Note that the first pass does not work like a phrase-based decoder because it yields dependency trees on the target side. A uniform model (i.e., each action has a fixed probability of 1/3) is used to resolve “h+h” conflicts.

system	MT02 (tune)		MT03		MT04		MT05	
	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
phrase	34.88	57.00	33.82	57.19	35.48	56.48	32.52	57.62
dependency	35.23	56.12	34.20	56.36	36.01	55.55	33.06	56.94
<i>this work</i>	35.71**	55.87**	34.81**+	55.94**+	36.37**	55.02**+	33.53**	56.58**

Table 2: Comparison with Moses (Koehn et al., 2007) and a re-implementation of the bottom-up string-to-dependency decoder (Shen et al., 2008) in terms of uncased BLEU and TER. We use randomization test (Riezler and Maxwell, 2005) to calculate statistical significance. \*: significantly better than Moses ( $p < 0.05$ ), \*\*: significantly better than Moses ( $p < 0.01$ ), +: significantly better than string-to-dependency ( $p < 0.05$ ), ++: significantly better than string-to-dependency ( $p < 0.01$ ).

features	BLEU	TER
standard	34.79	56.93
+ depLM	35.29*	56.17**
+ maxent	35.40**	56.09**
+ depLM & maxent	35.71**	55.87**

Table 3: Contribution of maximum entropy shift-reduce parsing model. “standard” denotes using standard features of phrase-based system. Adding dependency language model (“depLM”) and the maximum entropy shift-reduce parsing model (“maxent”) significantly improves BLEU and TER on the development set, both separately and jointly.

4-gram language model on the Xinhua portion of the GIGAWORD corpus, which contains 238M English words. A 3-gram dependency language model was trained on the English dependency trees. We used the 2002 NIST MT Chinese-English dataset as the development set and the 2003-2005 NIST datasets as the testsets. We evaluated translation quality using *uncased* BLEU (Papineni et al., 2002) and TER (Snover et al., 2006). The features were optimized with respect to BLEU using the minimum error rate training algorithm (Och, 2003).

We chose the following two systems that are closest to our work as baselines:

1. The Moses phrase-based decoder (Koehn et al., 2007).
2. A re-implementation of bottom-up string-to-dependency decoder (Shen et al., 2008).

All the three systems share with the same target-side parsed, word-aligned training data. The histogram pruning parameter  $b$  is set to 100 and

rules	coverage	BLEU	TER
well-formed	44.87	34.42	57.35
all	100.00	35.71**	55.87**

Table 4: Comparison of well-formed and ill-formed structures. Using all rules significantly outperforms using only well-formed structures. BLEU and TER scores are calculated on the development set.

phrase table limit is set to 20 for all the three systems. Moses shares the same feature set with our system except for the dependency features. For the bottom-up string-to-dependency system, we included both well-formed and ill-formed structures in chart parsing. To control the grammar size, we only extracted “tight” initial phrase pairs (i.e., the boundary words of a phrase must be aligned) as suggested by (Chiang, 2007). For our system, we used the Le Zhang’s maximum entropy modeling toolkit to train the shift-reduce parsing model after extracting 32.6M events from the training data.<sup>4</sup> We set the iteration limit to 100. The accuracy on the training data is 90.18%.

Table 2 gives the performance of Moses, the bottom-up string-to-dependency system, and our system in terms of uncased BLEU and TER scores. From the same training data, Moses extracted 103M bilingual phrases, the bottom-up string-to-dependency system extracted 587M string-to-dependency rules, and our system extracted 124M phrase-based dependency rules. We find that our approach outperforms both baselines systematically on all testsets. We use randomization test (Riezler and Maxwell, 2005) to calculate statistical significance. As our system can take full advantage of lexicalized reordering and depen-

<sup>4</sup><http://homepages.inf.ed.ac.uk/lzhang10/maxent.html>

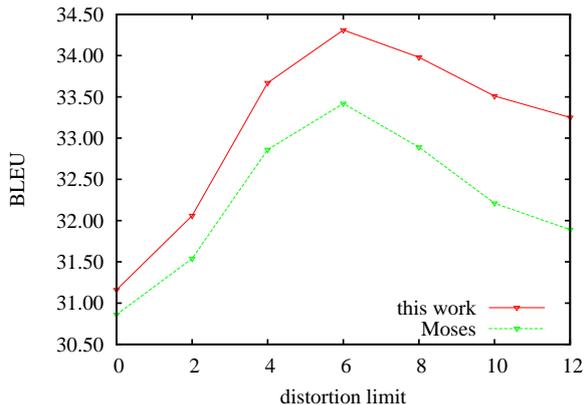


Figure 5: Performance of Moses and our system with various distortion limits.

dependency language models without loss in rule coverage, it achieves significantly better results than Moses on all test sets. The gains in TER are much larger than BLEU because dependency language models do not model  $n$ -grams directly. Compared with the bottom-up string-to-dependency system, our system outperforms consistently but not significantly in all cases. The average decoding time for Moses is 3.67 seconds per sentence, bottom-up string-to-dependency is 13.89 seconds, and our system is 4.56 seconds.

Table 3 shows the effect of hypergraph reranking. In the first pass, our decoder uses standard phrase-based features to build a hypergraph. The BLEU score is slightly lower than Moses with the same configuration. One possible reason is that our decoder organizes stacks with respect to actions, whereas Moses groups partial translations with the same number of covered source words in stacks. In the second pass, our decoder reranks the hypergraph with additional dependency features. We find that adding dependency language and maximum entropy shift-reduce models consistently brings significant improvements, both separately and jointly.

We analyzed translation rules extracted from the training data. Among them, well-formed structures account for 43.58% (fixed 33.21%, floating left 9.01%, and floating right 1.36%) and ill-formed structures 56.42%. As shown in Table 4, using all rules clearly outperforms using only well-formed structures.

Figure 5 shows the performance of Moses and our system with various distortion limits on the development set. Our system consistently outper-

forms Moses in all cases, suggesting that adding dependency helps improve phrase reordering.

## 6 Related Work

The work of Galley and Manning (2009) is closest in spirit to ours. They introduce maximum spanning tree (MST) parsing (McDonald et al., 2005) into phrase-based translation. The system is phrase-based except that an MST parser runs to parse partial translations at the same time. One challenge is that MST parsing itself is not incremental, making it expensive to identify loops during hypothesis expansion. On the contrary, shift-reduce parsing is naturally incremental and can be seamlessly integrated into left-to-right phrase-based decoding. More importantly, in our work dependency trees are memorized for phrases rather than being generated word by word on the fly in decoding. This treatment might not only reduce decoding complexity but also potentially revolve local parsing ambiguity.

Our decoding algorithm is similar to Gimpel and Smith (2011)’s lattice parsing algorithm as we divide decoding into two steps: hypergraph generation and hypergraph rescoring. The major difference is that our hypergraph is not a phrasal lattice because each phrase pair is associated with a dependency structure on the target side. In other words, our second pass is to find the Viterbi derivation with addition features rather than parsing the phrasal lattice. In addition, their algorithm produces phrasal dependency parse trees while the leaves of our dependency trees are words, making dependency language models can be directly used.

Shift-reduce parsing has been successfully used in phrase-based decoding but limited to adding structural constraints. Galley and Manning (2008) propose a shift-reduce algorithm to integrate a hierarchical reordering model into phrase-based systems. Feng et al. (2010) use shift-reduce parsing to impose ITG (Wu, 1997) constraints on phrase permutation. Our work differs from theirs by going further to incorporate linguistic syntax into phrase-based decoding.

Along another line, a number of authors have developed incremental algorithms for syntax-based models (Watanabe et al., 2006; Huang and Mi, 2010; Dyer and Resnik, 2010; Feng et al., 2012). Watanabe et al. (2006) introduce an Early-style top-down parser based on binary-branching Greibach Normal Form. Huang et al. (2010), Dyer

and Resnik (2010), and Feng et al. (2012) use dotted rules to change the tree transversal to generate target words left-to-right, either top-down or bottom-up.

## 7 Conclusion

We have presented a shift-reduce parsing algorithm for phrase-based string-to-dependency translation. The algorithm generates dependency structures incrementally using string-to-dependency phrase pairs. Therefore, our approach is capable of combining the advantages of both phrase-based and string-to-dependency models, it outperforms the two baselines on Chinese-to-English translation.

In the future, we plan to include more contextual information (e.g., the uncovered source phrases) in the maximum entropy model to resolve conflicts. Another direction is to adapt the dynamic programming algorithm proposed by Huang and Sagae (2010) to improve our string-to-dependency decoder. It is also interesting to compare with applying word-based shift-reduce parsing to phrase-based decoding similar to (Galley and Manning, 2009).

## Acknowledgments

This research is supported by the 863 Program under the grant No 2012AA011102 and No. 2011AA01A207, by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office, and by a Research Fund No. 20123000007 from Tsinghua MOE-Microsoft Joint Laboratory.

## References

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL 2005*.

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

Chris Dyer and Philip Resnik. 2010. Context-free reordering, finite-state translation. In *Proc. of NAACL 2010*.

Yang Feng, Haitao Mi, Yang Liu, and Qun Liu. 2010. An efficient shift-reduce decoding algorithm for phrase-based machine translation. In *Proc. of COLING 2010*.

Yang Feng, Yang Liu, Qun Liu, and Trevor Cohn. 2012. Left-to-right tree-to-string decoding with prediction. In *Proc. of EMNLP 2012*.

Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proc. of EMNLP 2008*.

Michel Galley and Christopher D. Manning. 2009. Quadratic-time dependency parsing for machine translation. In *Proc. of ACL 2009*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. of ACL 2006*.

Kevin Gimpel and Noah A. Smith. 2011. Quasi-synchronous phrase dependency grammars for machine translation. In *Proc. of EMNLP 2011*.

Liang Huang and David Chiang. 2007. Forest rescore: Faster decoding with integrated language models. In *Proc. of ACL 2007*.

Liang Huang and Haitao Mi. 2010. Efficient incremental decoding for tree-to-string translation. In *Proc. of EMNLP 2010*.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proc. of ACL 2010*.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. of AMTA 2006*.

Liang Huang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proc. of EMNLP 2009*.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proc. of ACL 2008*.

Dan Klein and Christopher Manning. 2003. Accurate unlexicalized parsing. In *Proc. of ACL 2003*.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*.

Philipp Koehn, Franz Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of NAACL 2003*.

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

- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. of ACL 2006*.
- Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. Spmt: Statistical machine translation with syntactified target language phrases. In *Proc. of EMNLP 2006*.
- R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of EMNLP 2005*.
- Haitao Mi and Liang Huang. 2008. Forest-based translation. In *Proc. of ACL 2008*.
- Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proc. of ACL 2004 Workshop Incremental Parsing: Bringing Engineering and Cognition Together*.
- Franz Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4).
- Franz Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL 2003*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL 2002*.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *Proc. of ACL 2005*.
- S. Riezler and J. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for mt. In *Proc. of ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proc. of ACL 2008*.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2010. String-to-dependency statistical machine translation. *Computational Linguistics*, 36(4).
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proc. of AMTA 2006*.
- Andreas Stolcke. 2002. Srlm - an extensible language modeling toolkit. In *Proc. of ICSLP 2002*.
- Taro Watanabe, Hajime Tsukuda, and Hideki Isozaki. 2006. Left-to-right target generation for hierarchical phrase-based translation. In *Proc. of ACL 2006*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL 2001*.
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam search. In *Proc. of EMNLP 2008*.
- Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proc. of ACL 2008*.

# Integrating Translation Memory into Phrase-Based Machine Translation during Decoding

Kun Wang<sup>†</sup>   Chengqing Zong<sup>†</sup>   Keh-Yih Su<sup>‡</sup>

<sup>†</sup>National Laboratory of Pattern Recognition, Institute of Automation,  
Chinese Academy of Sciences, Beijing, China

<sup>‡</sup>Behavior Design Corporation, Taiwan

<sup>†</sup>{kunwang, cqzong}@nlpr.ia.ac.cn, <sup>‡</sup>kysu@bdc.com.tw

## Abstract

Since statistical machine translation (SMT) and translation memory (TM) complement each other in matched and unmatched regions, integrated models are proposed in this paper to incorporate TM information into phrase-based SMT. Unlike previous multi-stage pipeline approaches, which directly merge TM result into the final output, the proposed models refer to the corresponding TM information associated with each phrase at SMT decoding. On a Chinese–English TM database, our experiments show that the proposed integrated Model-III is significantly better than either the SMT or the TM systems when the fuzzy match score is above 0.4. Furthermore, integrated Model-III achieves overall 3.48 BLEU points improvement and 2.62 TER points reduction in comparison with the pure SMT system. Besides, the proposed models also outperform previous approaches significantly.

## 1 Introduction

Statistical machine translation (SMT), especially the phrase-based model (Koehn et al., 2003), has developed very fast in the last decade. For certain language pairs and special applications, SMT output has reached an acceptable level, especially in the domains where abundant parallel corpora are available (He et al., 2010). However, SMT is rarely applied to professional translation because its output quality is still far from satisfactory. Especially, there is no guarantee that a SMT system can produce translations in a consistent manner (Ma et al., 2011).

In contrast, translation memory (TM), which uses the most similar translation sentence (usually above a certain fuzzy match threshold) in the database as the reference for post-editing, has

been widely adopted in professional translation field for many years (Lagoudaki, 2006). TM is very useful for repetitive material such as updated product manuals, and can give high quality and consistent translations when the similarity of fuzzy match is high. Therefore, professional translators trust TM much more than SMT. However, high-similarity fuzzy matches are available unless the material is very repetitive.

In general, for those matched segments<sup>1</sup>, TM provides more reliable results than SMT does. One reason is that the results of TM have been revised by human according to the global context, but SMT only utilizes local context. However, for those unmatched segments, SMT is more reliable. Since TM and SMT complement each other in those matched and unmatched segments, the output quality is expected to be raised significantly if they can be combined to supplement each other.

In recent years, some previous works have incorporated TM matched segments into SMT in a pipelined manner (Koehn and Senellart, 2010; Zhechev and van Genabith, 2010; He et al., 2011; Ma et al., 2011). All these pipeline approaches translate the sentence in two stages. They first determine whether the extracted TM sentence pair should be adopted or not. Most of them use fuzzy match score as the threshold, but He et al. (2011) and Ma et al. (2011) use a classifier to make the judgment. Afterwards, they merge the relevant translations of matched segments into the source sentence, and then force the SMT system to only translate those unmatched segments at decoding.

There are three obvious drawbacks for the above pipeline approaches. Firstly, all of them determine whether those matched segments

---

<sup>1</sup> We mean “sub-sentential segments” in this work.

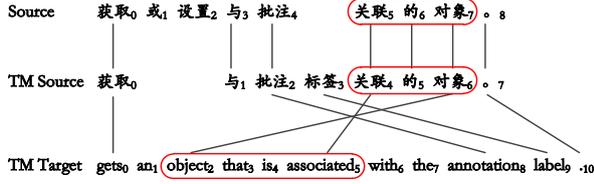


Figure 1: Phrase Mapping Example

should be adopted or not at sentence level. That is, they are either all adopted or all abandoned regardless of their individual quality. Secondly, as several TM target phrases might be available for one given TM source phrase due to insertions, the incorrect selection made in the merging stage cannot be remedied in the following translation stage. For example, there are six possible corresponding TM target phrases for the given TM source phrase “关联<sub>4</sub> 的<sub>5</sub> 对象<sub>6</sub>” (as shown in Figure 1) such as “object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>”, and “an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub> with<sub>6</sub>”, etc. And it is hard to tell which one should be adopted in the merging stage. Thirdly, the pipeline approach does not utilize the SMT probabilistic information in deciding whether a matched TM phrase should be adopted or not, and which target phrase should be selected when we have multiple candidates. Therefore, the possible improvements resulted from those pipeline approaches are quite limited.

On the other hand, instead of directly merging TM matched phrases into the source sentence, some approaches (Bi çici and Dymetman, 2008; Simard and Isabelle, 2009) simply add the longest matched pairs into SMT phrase table, and then associate them with a fixed large probability value to favor the corresponding TM target phrase at SMT decoding. However, since only one aligned target phrase will be added for each matched source phrase, they share most drawbacks with the pipeline approaches mentioned above and merely achieve similar performance.

To avoid the drawbacks of the pipeline approach (mainly due to making a hard decision *before* decoding), we propose several integrated models to completely make use of TM information *during* decoding. For each TM source phrase, we keep all its possible corresponding target phrases (instead of keeping only one of them). The integrated models then consider all corresponding TM target phrases and SMT preference during decoding. Therefore, the proposed integrated models combine SMT and TM at a deep level (versus the surface level at which TM result is directly plugged in under previous pipeline approaches).

On a Chinese–English computer technical documents TM database, our experiments have shown that the proposed Model-III improves the translation quality significantly over either the pure phrase-based SMT or the TM systems when the fuzzy match score is above 0.4. Compared with the pure SMT system, the proposed integrated Model-III achieves 3.48 BLEU points improvement and 2.62 TER points reduction overall. Furthermore, the proposed models significantly outperform previous pipeline approaches.

## 2 Problem Formulation

Compared with the standard phrase-based machine translation model, the translation problem is reformulated as follows (only based on the best TM, however, it is similar for multiple TM sentences):

$$\hat{t} = \arg \max_t P(t|s, [tm\_s, tm\_t, tm\_f, s\_a, tm\_a]) \quad (1)$$

Where  $s$  is the given source sentence to be translated,  $t$  is the corresponding target sentence and  $\hat{t}$  is the final translation;  $[tm\_s, tm\_t, tm\_f, s\_a, tm\_a]$  are the associated information of the best TM sentence-pair;  $tm\_s$  and  $tm\_t$  denote the corresponding TM sentence pair;  $tm\_f$  denotes its associated fuzzy match score (from 0.0 to 1.0);  $s\_a$  is the editing operations between  $tm\_s$  and  $s$ ; and  $tm\_a$  denotes the word alignment between  $tm\_s$  and  $tm\_t$ .

Let  $\bar{s}_{a(k)}$  and  $\bar{t}_k$  denote the  $k$ -th associated source phrase and target phrase, respectively. Also,  $\bar{s}_{a(1)}^{a(K)}$  and  $\bar{t}_1^K$  denote the associated source phrase sequence and the target phrase sequence, respectively (total  $K$  phrases without insertion). Then the above formula (1) can be decomposed as below:

$$\begin{aligned} \hat{t} &= \arg \max_t P(t|s, tm\_s, tm\_t, tm\_f, s\_a, tm\_a) \\ &= \arg \max_t \sum_{[\bar{s}_1^K = s, \bar{t}_1^K = t]} P(\bar{t}_1^K | \bar{s}_{a(1)}^{a(K)} | s, tm\_s, tm\_t, tm\_f, s\_a, tm\_a) \\ &\triangleq \arg \max_t \max_{[\bar{s}_1^K = s, \bar{t}_1^K = t]} \left\{ P(\bar{t}_1^K | \bar{s}_{a(1)}^{a(K)}, tm\_s, tm\_t, tm\_f, s\_a, tm\_a) \right\} \\ &\quad \times P(\bar{s}_1^K | s) \end{aligned} \quad (2)$$

Afterwards, for any given source phrase  $\bar{s}_{a(k)}$ , we can find its corresponding TM source phrase  $tm\_s_{\bar{s}_{a(k)}}$  and all possible TM target phrases (each of them is denoted by  $tm\_t_{\bar{t}_{a(k)}}$ ) with the help of corresponding editing operations  $s\_a$  and word alignment  $tm\_a$ . As mentioned above, we can have six different possible TM target phrases for the TM source phrase “关联<sub>4</sub> 的<sub>5</sub> 对象<sub>6</sub>”. This

is because there are insertions around the directly aligned TM target phrase.

In the above Equation (2), we first segment the given source sentence into various phrases, and then translate the sentence based on those source phrases. Also,  $\bar{s}_{a(1)}^{a(K)}$  is replaced by  $\bar{s}_1^K$ , as they are actually the same segmentation sequence. Assume that the segmentation probability  $P(\bar{s}_1^K|s)$  is a uniform distribution, with the corresponding TM source and target phrases obtained above, this problem can be further simplified as follows:

$$\begin{aligned}
& P(\bar{t}_1^K | \bar{s}_{a(1)}^{a(K)}, tm\_s, tm\_t, tm\_f, s\_a, tm\_a) \\
&= \sum_{tm\_t_{a(1)}^{a(K)}} P(\bar{t}_1^K, tm\_t_{a(1)}^{a(K)} | \bar{s}_{a(1)}^{a(K)}, tm\_s_{a(1)}^{a(K)}, tm\_t, z) \\
&\approx \max_{tm\_t_{a(1)}^{a(K)}} P(\bar{t}_1^K, tm\_t_{a(1)}^{a(K)} | \bar{s}_{a(1)}^{a(K)}, tm\_s_{a(1)}^{a(K)}, tm\_t, z) \\
&\approx \max_{tm\_t_{a(1)}^{a(K)}} P(\bar{t}_1^K, M_1^K | \bar{s}_{a(1)}^{a(K)}, L_1^K, z) \\
&\approx P(\bar{t}_1^K | \bar{s}_{a(1)}^{a(K)}) \times \prod_{k=1}^K \max_{tm\_t_{a(k)}} P(M_k | L_k, z)
\end{aligned} \tag{3}$$

Where  $M_k$  is the corresponding TM phrase matching status for  $\bar{t}_k$ , which is a vector consisting of various indicators (e.g., Target Phrase Content Matching Status, etc., to be defined later), and reflects the quality of the given candidate;  $L_k$  is the linking status vector of  $\bar{s}_{a(k)}$  (the aligned source phrase of  $\bar{t}_k$  within  $\bar{s}_1^K$ ), and indicates the matching and linking status in the source side (which is closely related to the status in the target side); also,  $z$  indicates the corresponding TM fuzzy match interval specified later.

In the second line of Equation (3), we convert the fuzzy match score  $tm\_f$  into its corresponding interval  $z$ , and incorporate all possible combinations of TM target phrases. Afterwards, we select the best one in the third line. Last, in the fourth line, we introduce the source matching status and the target linking status (detailed features would be defined later). Since we might have several possible TM target phrases  $tm\_t_{a(k)}$ , the one with the maximum score will be adopted during decoding.

The first factor  $P(\bar{t}_1^K | \bar{s}_{a(1)}^{a(K)})$  in the above formula (3) is just the typical phrase-based SMT model, and the second factor  $P(M_k | L_k, z)$  (to be specified in the Section 3) is the information derived from the TM sentence pair. Therefore, we can still keep the original phrase-based SMT model and only pay attention to how to extract

useful information from the best TM sentence pair to guide SMT decoding.

### 3 Proposed Models

Three integrated models are proposed to incorporate different features as follows:

#### 3.1 Model-I

In this simplest model, we only consider *Target Phrase Content Matching Status* (TCM) for  $M_k$ . For  $L_k$ , we consider four different features at the same time: *Source Phrase Content Matching Status* (SCM), *Number of Linking Neighbors* (NLN), *Source Phrase Length* (SPL), and *Sentence End Punctuation Indicator* (SEP). Those features will be defined below.  $P(M_k | L_k, z)$  is then specified as:

$$P(M_k | L_k, z) \triangleq P(TCM_k | SCM_k, NLN_k, SPL_k, SEP_k, z)$$

All features incorporated in this model are specified as follows:

**TM Fuzzy Match Interval (z):** The *fuzzy match score* (FMS) between source sentence  $s$  and TM source sentence  $tm\_s$  indicates the reliability of the given TM sentence, and is defined as (Sikes, 2007):

$$FMS(s, tm\_s) = 1 - \frac{\text{Levenshtein}(s, tm\_s)}{\max(|s|, |tm\_s|)}$$

Where  $\text{Levenshtein}(s, tm\_s)$  is the word-based Levenshtein Distance (Levenshtein, 1966) between  $s$  and  $tm\_s$ . We equally divide FMS into ten fuzzy match intervals such as: [0.9, 1.0), [0.8, 0.9) etc., and the index  $z$  specifies the corresponding interval. For example, since the fuzzy match score between  $s$  and  $tm\_s$  in Figure 1 is 0.667, then  $z = [0.6, 0.7)$ .

**Target Phrase Content Matching Status (TCM):** It indicates the content matching status between  $\bar{t}_k$  and  $tm\_t_{a(k)}$ , and reflects the quality of  $\bar{t}_k$ . Because  $tm\_t$  is nearly perfect when FMS is high, if the similarity between  $\bar{t}_k$  and  $tm\_t_{a(k)}$  is high, it implies that the given  $\bar{t}_k$  is possibly a good candidate. It is a member of  $\{Same, High, Low, NA (Not-Applicable)\}$ , and is specified as:

- (1) If  $tm\_t_{a(k)}$  is not null:
  - (a) if  $FMS(\bar{t}_k, tm\_t_{a(k)}) = 1.0$ ,  $TCM_k = Same$ ;
  - (b) else if  $FMS(\bar{t}_k, tm\_t_{a(k)}) > 0.5$ ,  $TCM_k = High$ ;
  - (c) else,  $TCM_k = Low$ ;
- (2) If  $tm\_t_{a(k)}$  is null,  $TCM_k = NA$ ;

Here  $tm\_t_{a(k)}$  is null means that either there is no corresponding TM source phrase  $tm\_s_{a(k)}$  or there is no corresponding TM target phrase

$tm_{\bar{t}_{a(k)}}$  aligned with  $tm_{\bar{s}_{a(k)}}$ . In the example of Figure 1, assume that the given  $\bar{s}_{a(k)}$  is “关联<sub>5</sub>的<sub>6</sub>对象<sub>7</sub>” and  $\bar{t}_k$  is “object that is associated”. If  $tm_{\bar{t}_{a(k)}}$  is “object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>”,  $TCM_k = Same$ ; if  $tm_{\bar{t}_{a(k)}}$  is “an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>”,  $TCM_k = High$ .

**Source Phrase Content Matching Status (SCM):** Which indicates the content matching status between  $\bar{s}_{a(k)}$  and  $tm_{\bar{s}_{a(k)}}$ , and it affects the matching status of  $\bar{t}_k$  and  $tm_{\bar{t}_{a(k)}}$  greatly. The more similar  $\bar{s}_{a(k)}$  is to  $tm_{\bar{s}_{a(k)}}$ , the more similar  $\bar{t}_k$  is to  $tm_{\bar{t}_{a(k)}}$ . It is a member of  $\{Same, High, Low, NA\}$  and is defined as:

- (1) If  $tm_{\bar{s}_{a(k)}}$  is not null:
  - (a) if  $FMS(\bar{s}_{a(k)}, tm_{\bar{s}_{a(k)}}) = 1.0$ ,  $SCM_k = Same$ ;
  - (b) else if  $FMS(\bar{s}_{a(k)}, tm_{\bar{s}_{a(k)}}) > 0.5$ ,  $SCM_k = High$ ;
  - (c) else,  $SCM_k = Low$ ;
- (2) If  $tm_{\bar{s}_{a(k)}}$  is null,  $SCM_k = NA$ ;

Here  $tm_{\bar{s}_{a(k)}}$  is null means that there is no corresponding TM source phrase  $tm_{\bar{s}_{a(k)}}$  for the given source phrase  $\bar{s}_{a(k)}$ . Take the source phrase  $\bar{s}_{a(k)}$  “关联<sub>5</sub>的<sub>6</sub>对象<sub>7</sub>” in Figure 1 for an example, since its corresponding  $tm_{\bar{s}_{a(k)}}$  is “关联<sub>4</sub>的<sub>5</sub>对象<sub>6</sub>”, then  $SCM_k = Same$ .

**Number of Linking Neighbors (NLN):** Usually, the context of a source phrase would affect its target translation. The more similar the context are, the more likely that the translations are the same. Therefore, this NLN feature reflects the number of matched neighbors (words) and it is a vector of  $\langle x, y \rangle$ . Where “x” denotes the number of matched source neighbors; and “y” denotes how many those neighbors are also linked to target words (not null), which also affects the TM target phrase selection. This feature is a member of  $\{\langle x, y \rangle: \langle 2, 2 \rangle, \langle 2, 1 \rangle, \langle 2, 0 \rangle, \langle 1, 1 \rangle, \langle 1, 0 \rangle, \langle 0, 0 \rangle\}$ . For the source phrase “关联<sub>5</sub>的<sub>6</sub>对象<sub>7</sub>” in Figure 1, the corresponding TM source phrase is “关联<sub>4</sub>的<sub>5</sub>对象<sub>6</sub>”. As only their right neighbors “。8” and “。7” are matched, and “。7” is aligned with “.10”, NLN will be  $\langle 1, 1 \rangle$ .

**Source Phrase Length (SPL):** Usually the longer the source phrase is, the more reliable the TM target phrase is. For example, the corresponding  $tm_{\bar{t}_{a(k)}}$  for the source phrase with 5 words would be more reliable than that with only one word. This feature denotes the number of words included in  $\bar{s}_{a(k)}$ , and is a member of  $\{1, 2, 3, 4, \geq 5\}$ . For the case “关联<sub>5</sub>的<sub>6</sub>对象<sub>7</sub>”, SPL will be 3.

**Sentence End Punctuation Indicator (SEP):** Which indicates whether the current phrase is a punctuation at the end of the sentence, and is a member of  $\{Yes, No\}$ . For example, the SEP for “关联<sub>5</sub>的<sub>6</sub>对象<sub>7</sub>” will be “No”. It is introduced because the SCM and TCM for a sentence-end-punctuation are always “Same” regardless of other features. Therefore, it is used to distinguish this special case from other cases.

### 3.2 Model-II

As Model-I ignores the relationship among various possible TM target phrases, we add two features *TM Candidate Set Status* (CSS) and *Longest TM Candidate Indicator* (LTC) to incorporate this relationship among them. Since CSS is redundant after LTC is known, we thus ignore it for evaluating TCM probability in the following derivation:

$$\begin{aligned}
P(M_k|L_k, z) &\triangleq P(TCM_k, LTC_k|SCM_k, NLN_k, CSS_k, SPL_k, SEP_k, z) \\
&\approx \left\{ \begin{array}{l} P(TCM_k|SCM_k, NLN_k, LTC_k, SPL_k, SEP_k, z) \\ \times P(LTC_k|CSS_k, SCM_k, NLN_k, SEP_k, z) \end{array} \right\}
\end{aligned}$$

The two new features CSS and LTC adopted in Model-II are defined as follows:

**TM Candidate Set Status (CSS):** Which restricts the possible status of  $tm_{\bar{t}_{a(k)}}$ , and is a member of  $\{Single, Left-Ext, Right-Ext, Both-Ext, NA\}$ . Where “Single” means that there is only one  $tm_{\bar{t}_{a(k)}}$  candidate for the given source phrase  $tm_{\bar{s}_{a(k)}}$ ; “Left-Ext” means that there are multiple  $tm_{\bar{t}_{a(k)}}$  candidates, and all the candidates are generated by extending only the left boundary; “Right-Ext” means that there are multiple  $tm_{\bar{t}_{a(k)}}$  candidates, and all the candidates are generated by only extending to the right; “Both-Ext” means that there are multiple  $tm_{\bar{t}_{a(k)}}$  candidates, and the candidates are generated by extending to both sides; “NA” means that  $tm_{\bar{t}_{a(k)}}$  is null.

For “关联<sub>4</sub>的<sub>5</sub>对象<sub>6</sub>” in Figure 1, the linked TM target phrase is “object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>”, and there are 5 other candidates by extending to both sides. Therefore,  $CSS_k = Both-Ext$ .

**Longest TM Candidate Indicator (LTC):** Which indicates whether the given  $tm_{\bar{t}_{a(k)}}$  is the longest candidate or not, and is a member of  $\{Original, Left-Longest, Right-Longest, Both-Longest, Medium, NA\}$ . Where “Original” means that the given  $tm_{\bar{t}_{a(k)}}$  is the one without extension; “Left-Longest” means that the given

$tm_{\bar{t}_{a(k)}}$  is only extended to the left and is the longest one; “*Right-Longest*” means that the given  $tm_{\bar{t}_{a(k)}}$  is only extended to the right and is the longest one; “*Both-Longest*” means that the given  $tm_{\bar{t}_{a(k)}}$  is extended to both sides and is the longest one; “*Medium*” means that the given  $tm_{\bar{t}_{a(k)}}$  has been extended but not the longest one; “*NA*” means that  $tm_{\bar{t}_{a(k)}}$  is null.

For  $tm_{\bar{t}_{a(k)}}$  “*object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*” in Figure 1,  $LTC_k = Original$ ; for  $tm_{\bar{t}_{a(k)}}$  “*an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*”,  $LTC_k = Left-Longest$ ; for the longest  $tm_{\bar{t}_{a(k)}}$  “*an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub> with<sub>6</sub> the<sub>7</sub>*”,  $LTC_k = Both-Longest$ .

### 3.3 Model-III

The abovementioned integrated models ignore the reordering information implied by TM. Therefore, we add a new feature *Target Phrase Adjacent Candidate Relative Position Matching Status* (CPM) into Model-II and Model-III is given as:

$$P(M_k|L_k, z) \triangleq P([TCM, LTC, CPM]_k | [SCM, NLN, CSS, SPL, SEP]_k, z) \approx \left\{ \begin{array}{l} P(TCM_k | SCM_k, NLN_k, LTC_k, SPL_k, SEP_k, z) \\ \times P(LTC_k | CSS_k, SCM_k, NLN_k, SEP_k, z) \\ \times P(CPM_k | TCM_k, SCM_k, NLN_k, z) \end{array} \right\}$$

We assume that CPM is independent with SPL and SEP, because the length of source phrase would not affect reordering too much and SEP is used to distinguish the sentence end punctuation with other phrases.

The new feature CPM adopted in Model-III is defined as:

**Target Phrase Adjacent Candidate Relative Position Matching Status (CPM):** Which indicates the matching status between the relative position of  $[\bar{t}_{k-1}, \bar{t}_k]$  and the relative position of  $[tm_{\bar{t}_{a(k-1)}}, tm_{\bar{t}_{a(k)}}]$ . It checks if  $[\bar{t}_{k-1}, \bar{t}_k]$  are positioned in the same order with  $[tm_{\bar{t}_{a(k-1)}}, tm_{\bar{t}_{a(k)}}]$ , and reflects the quality of ordering the given target candidate  $\bar{t}_k$ . It is a member of  $\{Adjacent-Same, Adjacent-Substitute, Linked-Interleaved, Linked-Cross, Linked-Reversed, Skip-Forward, Skip-Cross, Skip-Reversed, NA\}$ . Recall that  $\bar{t}_k$  is always right adjacent to  $\bar{t}_{k-1}$ , then various cases are defined as follows:

- (1) If both  $tm_{\bar{t}_{a(k-1)}}$  and  $tm_{\bar{t}_{a(k)}}$  are not null:
  - (a) If  $tm_{\bar{t}_{a(k)}}$  is on the right of  $tm_{\bar{t}_{a(k-1)}}$  and they are also adjacent to each other:
    - i. If the right boundary words of  $\bar{t}_{k-1}$  and  $tm_{\bar{t}_{a(k-1)}}$  are the same, and the left

boundary words of  $\bar{t}_k$  and  $tm_{\bar{t}_{a(k)}}$  are the same,  $CPM_k = Adjacent-Same$ ;

ii. Otherwise,  $CPM_k = Adjacent-Substitute$ ;

- (b) If  $tm_{\bar{t}_{a(k)}}$  is on the right of  $tm_{\bar{t}_{a(k-1)}}$  but they are not adjacent to each other,  $CPM_k = Linked-Interleaved$ ;

- (c) If  $tm_{\bar{t}_{a(k)}}$  is not on the right of  $tm_{\bar{t}_{a(k-1)}}$ :
  - i. If there are cross parts between  $tm_{\bar{t}_{a(k)}}$  and  $tm_{\bar{t}_{a(k-1)}}$ ,  $CPM_k = Linked-Cross$ ;

ii. Otherwise,  $CPM_k = Linked-Reversed$ ;

- (2) If  $tm_{\bar{t}_{a(k-1)}}$  is null but  $tm_{\bar{t}_{a(k)}}$  is not null, then find the first  $tm_{\bar{t}_{a(k-n)}} (k \geq n)$  which is not null ( $n$  starts from 2)<sup>2</sup>:
  - (a) If  $tm_{\bar{t}_{a(k)}}$  is on the right of  $tm_{\bar{t}_{a(k-n)}}$ ,  $CPM_k = Skip-Forward$ ;

- (b) If  $tm_{\bar{t}_{a(k)}}$  is not on the right of  $tm_{\bar{t}_{a(k-n)}}$ :
  - i. If there are cross parts between  $tm_{\bar{t}_{a(k)}}$  and  $tm_{\bar{t}_{a(k-n)}}$ ,  $CPM_k = Skip-Cross$ ;

ii. Otherwise,  $CPM_k = Skip-Reversed$ .

- (3) If  $tm_{\bar{t}_{a(k)}}$  is null,  $CPM_k = NA$ .

In Figure 1, assume that  $\bar{t}_{k-1}$ ,  $\bar{t}_k$  and  $tm_{\bar{t}_{a(k-1)}}$  are “*gets an*”, “*object that is associated with*” and “*gets<sub>0</sub> an<sub>1</sub>*”, respectively. For  $tm_{\bar{t}_{a(k)}}$  “*object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*”, because  $tm_{\bar{t}_{a(k)}}$  is on the right of  $tm_{\bar{t}_{a(k-1)}}$  and they are adjacent pair, and both boundary words (“*an*” and “*an<sub>1</sub>*”; “*object*” and “*object<sub>2</sub>*”) are matched,  $CPM_k = Adjacent-Same$ ; for  $tm_{\bar{t}_{a(k)}}$  “*an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*”, because there are cross parts “*an<sub>1</sub>*” between  $tm_{\bar{t}_{a(k)}}$  and  $tm_{\bar{t}_{a(k-1)}}$ ,  $CPM_k = Linked-Cross$ . On the other hand, assume that  $\bar{t}_{k-1}$ ,  $\bar{t}_k$  and  $tm_{\bar{t}_{a(k-1)}}$  are “*gets*”, “*object that is associated with*” and “*gets<sub>0</sub>*”, respectively. For  $tm_{\bar{t}_{a(k)}}$  “*an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*”, because  $tm_{\bar{t}_{a(k)}}$  and  $tm_{\bar{t}_{a(k-1)}}$  are adjacent pair, but the left boundary words of  $\bar{t}_k$  and  $tm_{\bar{t}_{a(k)}}$  (“*object*” and “*an<sub>1</sub>*”) are not matched,  $CPM_k = Adjacent-Substitute$ ; for  $tm_{\bar{t}_{a(k)}}$  “*object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*”, because  $tm_{\bar{t}_{a(k)}}$  is on the right of  $tm_{\bar{t}_{a(k-1)}}$  but they are not adjacent pair, therefore,  $CPM_k = Linked-Interleaved$ . One more example, assume that  $\bar{t}_{k-1}$ ,  $\bar{t}_k$  and  $tm_{\bar{t}_{a(k-1)}}$  are “*the annotation label*”, “*object that is associated with*” and “*the<sub>7</sub> annotation<sub>8</sub> label<sub>9</sub>*”, respectively. For  $tm_{\bar{t}_{a(k)}}$  “*an<sub>1</sub> object<sub>2</sub> that<sub>3</sub> is<sub>4</sub> associated<sub>5</sub>*”, because  $tm_{\bar{t}_{a(k)}}$  is on the left of  $tm_{\bar{t}_{a(k-1)}}$ , and there are no cross parts,  $CPM_k = Linked-Reversed$ .

<sup>2</sup> It can be identified by simply memorizing the index of nearest non-null  $tm_{\bar{t}_{a(k-n)}}$  during search.

## 4 Experiments

### 4.1 Experimental Setup

Our TM database consists of computer domain Chinese-English translation sentence-pairs, which contains about 267k sentence-pairs. The average length of Chinese sentences is 13.85 words and that of English sentences is 13.86 words. We randomly selected a development set and a test set, and then the remaining sentence pairs are for training set. The detailed corpus statistics are shown in Table 1. Furthermore, development set and test set are divided into various intervals according to their best fuzzy match scores. Corpus statistics for each interval in the test set are shown in Table 2.

For the phrase-based SMT system, we adopted the Moses toolkit (Koehn et al., 2007). The system configurations are as follows: GIZA++ (Och and Ney, 2003) is used to obtain the bidirectional word alignments. Afterwards, “intersection”<sup>3</sup> refinement (Koehn et al., 2003) is adopted to extract phrase-pairs. We use the SRI Language Model toolkit (Stolcke, 2002) to train a 5-gram model with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998) on the target-side (English) training corpus. All the feature weights and the weight for each probability factor (3 factors for Model-III) are tuned on the development set with minimum-error-rate training (MERT) (Och, 2003). The maximum phrase length is set to 7 in our experiments.

In this work, the translation performance is measured with case-insensitive BLEU-4 score (Papineni et al., 2002) and TER score (Snover et al., 2006). Statistical significance test is conducted with re-sampling (1,000 times) approach (Koehn, 2004) in 95% confidence level.

### 4.2 Cross-Fold Translation

To estimate the probabilities of proposed models, the corresponding phrase segmentations for bilingual sentences are required. As we want to check what actually happened during decoding in the real situation, cross-fold translation is used to obtain the corresponding phrase segmentations. We first extract 95% of the bilingual sentences as a new training corpus to train a SMT system. Afterwards, we generate the corresponding phrase segmentations for the remaining 5% bi-

<sup>3</sup> “grow-diag-final” and “grow-diag-final-and” are also tested. However, “intersection” is the best option in our experiments, especially for those high fuzzy match intervals.

	Train	Develop	Test
#Sentences	261,906	2,569	2,576
#Chn. Words	3,623,516	38,585	38,648
#Chn. VOC.	43,112	3,287	3,460
#Eng. Words	3,627,028	38,329	38,510
#Eng. VOC.	44,221	3,993	4,046

Table 1: Corpus Statistics

Intervals	#Sentences	#Words	W/S
[0.9, 1.0)	269	4,468	16.6
[0.8, 0.9)	362	5,004	13.8
[0.7, 0.8)	290	4,046	14.0
[0.6, 0.7)	379	4,998	13.2
[0.5, 0.6)	472	6,073	12.9
[0.4, 0.5)	401	5,921	14.8
[0.3, 0.4)	305	5,499	18.0
(0.0, 0.3)	98	2,639	26.9
<b>(0.0, 1.0)</b>	<b>2,576</b>	<b>38,648</b>	<b>15.0</b>

Table 2: Corpus Statistics for Test-Set

lingual sentences with *Forced Decoding* (Li et al., 2000; Zollmann et al., 2008; Auli et al., 2009; Wisniewski et al., 2010), which searches the best phrase segmentation for the specified output. Having repeated the above steps 20 times<sup>4</sup>, we obtain the corresponding phrase segmentations for the SMT training data (which will then be used to train the integrated models).

Due to OOV words and insertion words, not all given source sentences can generate the desired results through forced decoding. Fortunately, in our work, 71.7% of the training bilingual sentences can generate the corresponding target results. The remaining 28.3% of the sentence pairs are thus not adopted for generating training samples. Furthermore, more than 90% obtained source phrases are observed to be less than 5 words, which explains why five different quantization levels are adopted for Source Phrase Length (SPL) in section 3.1.

### 4.3 Translation Results

After obtaining all the training samples via cross-fold translation, we use Factored Language Model toolkit (Kirchhoff et al., 2007) to estimate the probabilities of integrated models with Witten-Bell smoothing (Bell et al., 1990; Witten et al., 1991) and Back-off method. Afterwards, we incorporate the TM information  $P(M_k|L_k, z)$  for each phrase at decoding. All experiments are

<sup>4</sup> This training process only took about 10 hours on our Ubuntu server (Intel 4-core Xeon 3.47GHz, 132 GB of RAM).

Intervals	TM	SMT	Model-I	Model-II	Model-III	Koehn-10	Ma-11	Ma-11-U
[0.9, 1.0)	81.31	81.38	85.44 *	86.47 **	<b>89.41</b> **	82.79	77.72	82.78
[0.8, 0.9)	73.25	76.16	79.97 *	80.89 *	<b>84.04</b> **	79.74 *	73.00	77.66
[0.7, 0.8)	63.62	67.71	71.65 *	72.39 *	<b>74.73</b> **	71.02 *	66.54	69.78
[0.6, 0.7)	43.64	54.56	54.88 #	55.88 **	<b>57.53</b> **	53.06	54.00	56.37
[0.5, 0.6)	27.37	46.32	47.32 **	47.45 **	<b>47.54</b> **	39.31	46.06	47.73
[0.4, 0.5)	15.43	37.18	37.25 #	37.60 #	<b>38.18</b> **	28.99	36.23	37.93
[0.3, 0.4)	8.24	29.27	<b>29.52</b> #	29.38 #	29.15 #	23.58	29.40	30.20
(0.0, 0.3)	4.13	<b>26.38</b>	25.61 #	25.32 #	25.57 #	18.56	26.30	26.92
<b>(0.0, 1.0)</b>	40.17	53.03	54.57 **	55.10 **	<b>56.51</b> **	50.31	51.98	54.32

Table 3: Translation Results (BLEU%). Scores marked by “\*” are significantly better ( $p < 0.05$ ) than both TM and SMT systems, and those marked by “#” are significantly better ( $p < 0.05$ ) than Koehn-10.

Intervals	TM	SMT	Model-I	Model-II	Model-III	Koehn-10	Ma-11	Ma-11-U
[0.9, 1.0)	9.79	13.01	9.22 #	8.52 **	<b>6.77</b> **	13.01	18.80	11.90
[0.8, 0.9)	16.21	16.07	13.12 **	12.74 **	<b>10.75</b> **	15.27	20.60	14.74
[0.7, 0.8)	27.79	22.80	19.10 **	18.58 **	<b>17.11</b> **	21.85	25.33	21.11
[0.6, 0.7)	46.40	33.38	32.63 #	32.27 **	<b>29.96</b> **	35.93	35.24	31.76
[0.5, 0.6)	62.59	39.56	<b>38.24</b> **	38.77 **	38.74 **	47.37	40.24	38.01
[0.4, 0.5)	73.93	47.19	47.03 #	46.34 **	<b>46.00</b> **	56.84	48.74	46.10
[0.3, 0.4)	79.86	55.71	<b>55.38</b> #	55.44 #	55.87 #	64.55	55.93	54.15
(0.0, 0.3)	85.31	<b>61.76</b>	62.38 #	63.66 #	63.51 #	73.30	63.00	60.67
<b>(0.0, 1.0)</b>	50.51	35.88	34.34 **	34.18 **	<b>33.26</b> **	40.75	38.10	34.49

Table 4: Translation Results (TER%). Scores marked by “\*” are significantly better ( $p < 0.05$ ) than both TM and SMT systems, and those marked by “#” are significantly better ( $p < 0.05$ ) than Koehn-10.

conducted using the Moses phrase-based decoder (Koehn et al., 2007).

Table 3 and 4 give the translation results of TM, SMT, and three integrated models in the test set. In the tables, the best translation results (either in BLEU or TER) at each interval have been marked in bold. Scores marked by “\*” are significantly better ( $p < 0.05$ ) than both the TM and the SMT systems.

It can be seen that TM significantly exceeds SMT at the interval [0.9, 1.0) in TER score, which illustrates why professional translators prefer TM rather than SMT as their assistant tool. Compared with TM and SMT, Model-I is significantly better than the SMT system in either BLEU or TER when the fuzzy match score is above 0.7; Model-II significantly outperforms both the TM and the SMT systems in either BLEU or TER when the fuzzy match score is above 0.5; Model-III significantly exceeds both the TM and the SMT systems in either BLEU or TER when the fuzzy match score is above 0.4. All these improvements show that our integrated models have combined the strength of both TM and SMT.

However, the improvements from integrated models get less when the fuzzy match score decreases. For example, Model-III outperforms

SMT 8.03 BLEU points at interval [0.9, 1.0), while the advantage is only 2.97 BLEU points at interval [0.6, 0.7). This is because lower fuzzy match score means that there are more unmatched parts between  $s$  and  $tm_s$ ; the output of TM is thus less reliable.

Across all intervals (the last row in the table), Model-III not only achieves the best BLEU score (56.51), but also gets the best TER score (33.26). If intervals are evaluated separately, when the fuzzy match score is above 0.4, Model-III outperforms both Model-II and Model-I in either BLEU or TER. Model-II also exceeds Model-I in either BLEU or TER. The only exception is at interval [0.5, 0.6), in which Model-I achieves the best TER score. This might be due to that the optimization criterion for MERT is BLEU rather than TER in our work.

#### 4.4 Comparison with Previous Work

In order to compare our proposed models with previous work, we re-implement two XML-Markup approaches: (Koehn and Senellart, 2010) and (Ma et al, 2011), which are denoted as **Koehn-10** and **Ma-11**, respectively. They are selected because they report superior performances in the literature. A brief description of them is as follows:

<b>Source</b>	如果 <sub>0</sub> 禁用 <sub>1</sub> 此 <sub>2</sub> 策略 <sub>3</sub> 设置 <sub>4</sub> ， <sub>5</sub> internet <sub>6</sub> explorer <sub>7</sub> 不 <sub>8</sub> 搜索 <sub>9</sub> internet <sub>10</sub> 查找 <sub>11</sub> 浏览器 <sub>12</sub> 的 <sub>13</sub> 新 <sub>14</sub> 版本 <sub>15</sub> ， <sub>16</sub> 因此 <sub>17</sub> 不 <sub>18</sub> 会 <sub>19</sub> 提示 <sub>20</sub> 用户 <sub>21</sub> 安装 <sub>22</sub> 。 <sub>23</sub>
<b>Reference</b>	if <sub>0</sub> you <sub>1</sub> disable <sub>2</sub> this <sub>3</sub> policy <sub>4</sub> setting <sub>5</sub> , <sub>6</sub> internet <sub>7</sub> explorer <sub>8</sub> does <sub>9</sub> not <sub>10</sub> check <sub>11</sub> the <sub>12</sub> internet <sub>13</sub> for <sub>14</sub> new <sub>15</sub> versions <sub>16</sub> of <sub>17</sub> the <sub>18</sub> browser <sub>19</sub> , <sub>20</sub> so <sub>21</sub> does <sub>22</sub> not <sub>23</sub> prompt <sub>24</sub> users <sub>25</sub> to <sub>26</sub> install <sub>27</sub> them <sub>28</sub> . <sub>29</sub>
<b>TM Source</b>	如果 <sub>0</sub> 不 <sub>1</sub> 配置 <sub>2</sub> 此 <sub>3</sub> 策略 <sub>4</sub> 设置 <sub>5</sub> ， <sub>6</sub> internet <sub>7</sub> explorer <sub>8</sub> 不 <sub>9</sub> 搜索 <sub>10</sub> internet <sub>11</sub> 查找 <sub>12</sub> 浏览器 <sub>13</sub> 的 <sub>14</sub> 新 <sub>15</sub> 版本 <sub>16</sub> ， <sub>17</sub> 因此 <sub>18</sub> 不 <sub>19</sub> 会 <sub>20</sub> 提示 <sub>21</sub> 用户 <sub>22</sub> 安装 <sub>23</sub> 。 <sub>24</sub>
<b>TM Target</b>	if <sub>0</sub> you <sub>1</sub> do <sub>2</sub> not <sub>3</sub> configure <sub>4</sub> this <sub>5</sub> policy <sub>6</sub> setting <sub>7</sub> , <sub>8</sub> internet <sub>9</sub> explorer <sub>10</sub> does <sub>11</sub> not <sub>12</sub> check <sub>13</sub> the <sub>14</sub> internet <sub>15</sub> for <sub>16</sub> new <sub>17</sub> versions <sub>18</sub> of <sub>19</sub> the <sub>20</sub> browser <sub>21</sub> , <sub>22</sub> so <sub>23</sub> does <sub>24</sub> not <sub>25</sub> prompt <sub>26</sub> users <sub>27</sub> to <sub>28</sub> install <sub>29</sub> them <sub>30</sub> . <sub>31</sub>
<b>TM Alignment</b>	0-0 1-3 2-4 3-5 4-6 5-7 6-8 7-9 8-10 9-11 11-15 13-21 14-19 15-17 16-18 17-22 18-23 19-24 21-26 22-27 23-29 24-31
<b>SMT</b>	if you disable this policy setting , internet explorer does not <i>prompt users to install</i> internet for new versions of the browser . [Miss 7 target words: 9~12, 20~21, 28; Has one wrong permutation]
<b>Koehn-10</b>	if you do you disable this policy setting , internet explorer does not check the internet for new versions of the browser , so does not prompt users to install them . [Insert two spurious target words]
<b>Ma-11</b>	if you disable this policy setting , internet explorer does not <i>prompt users to install</i> internet for new versions of the browser . [Miss 7 target words: 9~12, 20~21, 28; Has one wrong permutation]
<b>Model-I</b>	if you disable this policy setting , internet explorer does not <i>prompt users to install</i> new versions of the browser , so does not <i>check the internet</i> . [Miss 2 target words: 14, 28; Has one wrong permutation]
<b>Model-II</b>	if you disable this policy setting , internet explorer does not <i>prompt users to install</i> new versions of the browser , so does not <i>check the internet</i> . [Miss 2 target words: 14, 28; Has one wrong permutation]
<b>Model-III</b>	if you disable this policy setting , internet explorer does not check the internet for new versions of the browser , so does not prompt users to install them . [Exactly the same as the reference]

Figure 2: A Translation Example at Interval [0.9, 1.0] (with FMS=0.920)

Koehn et al. (2010) first find out the unmatched parts between the given source sentence and TM source sentence. Afterwards, for each *unmatched* phrase in the TM source sentence, they replace its corresponding translation in the *TM target sentence* by the corresponding *source phrase* in the input sentence, and then mark the substitution part. After replacing the corresponding translations of all unmatched source phrases in the TM target sentence, an XML input sentence (with mixed TM target phrases and marked input source phrases) is thus obtained. The SMT decoder then only translates the unmatched/marked source phrases and gets the desired results. Therefore, the *inserted* parts in the TM target sentence are automatically *included*. They use fuzzy match score to determine whether the current sentence should be marked or not; and their experiments show that this method is only effective when the fuzzy match score is above 0.8.

Ma et al. (2011) think fuzzy match score is not reliable and use a discriminative learning method to decide whether the current sentence should be

marked or not. Another difference between Ma-11 and Koehn-10 is how the XML input is constructed. In constructing the XML input sentence, Ma-11 replaces each *matched* source phrase in the *given source sentence* with the corresponding TM target phrase. Therefore, the *inserted* parts in the TM target sentence are *not included*. In Ma’s another paper (He et al., 2011), more linguistic features for discriminative learning are also added. In our work, we only re-implement the XML-Markup method used in (He et al., 2011; Ma et al., 2011), but do not implement the discriminative learning method. This is because the features adopted in their discriminative learning are complicated and difficult to re-implement. However, the proposed Model-III even outperforms the upper bound of their methods, which will be discussed later.

Table 3 and 4 give the translation results of Koehn-10 and Ma-11 (without the discriminator). Scores marked by “#” are significantly better ( $p < 0.05$ ) than Koehn-10. Besides, the upper bound of (Ma et al, 2011) is also given in the tables, which is denoted as **Ma-11-U**. We calculate this

upper bound according to the method described in (Ma et al., 2011). Since He et al., (2011) only add more linguistic features to the discriminative learning method, the upper bound of (He et al., 2011) is still the same with (Ma et al., 2011); therefore, Ma-11-U applies for both cases.

It is observed that Model-III significantly exceeds Koehn-10 at all intervals. More importantly, the proposed models achieve much better TER score than the TM system does at interval [0.9, 1.0), but Koehn-10 does not even exceed the TM system at this interval. Furthermore, Model-III is much better than Ma-11-U at most intervals. Therefore, it can be concluded that the proposed models outperform the pipeline approaches significantly.

Figure 2 gives an example at interval [0.9, 1.0), which shows the difference among different system outputs. It can be seen that “you do” is redundant for Koehn-10, because they are insertions and thus are kept in the XML input. However, SMT system still inserts another “you”, regardless of “you do” has already existed. This problem does not occur at Ma-11, but it misses some words and adopts one wrong permutation. Besides, Model-I selects more right words than SMT does but still puts them in wrong positions due to ignoring TM reordering information. In this example, Model-II obtains the same results with Model-I because it also lacks reordering information. Last, since Model-III considers both TM content and TM position information, it gives a perfect translation.

## 5 Conclusion and Future Work

Unlike the previous pipeline approaches, which directly merge TM phrases into the final translation result, we integrate TM information of each source phrase into the phrase-based SMT at decoding. In addition, all possible TM target phrases are kept and the proposed models select the best one during decoding via referring SMT information. Besides, the integrated model considers the probability information of both SMT and TM factors.

The experiments show that the proposed Model-III outperforms both the TM and the SMT systems significantly ( $p < 0.05$ ) in either BLEU or TER when fuzzy match score is above 0.4. Compared with the pure SMT system, Model-III achieves overall 3.48 BLEU points improvement and 2.62 TER points reduction on a Chinese-English TM database. Furthermore, Model-III significantly exceeds all previous pipeline ap-

proaches. Similar improvements are also observed on the Hansards parts of LDC2004T08 (not shown in this paper due to space limitation). Since no language-dependent feature is adopted, the proposed approaches can be easily adapted for other language pairs.

Moreover, following the approaches of Koehn-10 and Ma-11 (to give a fair comparison), training data for SMT and TM are the same in the current experiments. However, the TM is expected to play an even more important role when the SMT training-set differs from the TM database, as additional phrase-pairs that are unseen in the SMT phrase table can be extracted from TM (which can then be dynamically added into the SMT phrase table at decoding time). Our another study has shown that the integrated model would be even more effective when the TM database and the SMT training data-set are from different corpora in the same domain (not shown in this paper). In addition, more source phrases can be matched if a set of high-FMS sentences, instead of only the sentence with the highest FMS, can be extracted and referred at the same time. And it could further raise the performance.

Last, some related approaches (Smith and Clark, 2009; Phillips, 2011) combine SMT and example-based machine translation (EBMT) (Nagao, 1984). It would be also interesting to compare our integrated approach with that of theirs.

## Acknowledgments

The research work has been funded by the Hi-Tech Research and Development Program (“863” Program) of China under Grant No. 2011AA01A207, 2012AA011101, and 2012AA011102 and also supported by the Key Project of Knowledge Innovation Program of Chinese Academy of Sciences under Grant No.KGZD-EW-501.

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions. Our sincere thanks are also extended to Dr. Yanjun Ma and Dr. Yifan He for their valuable discussions during this study.

## References

- Michael Auli, Adam Lopez, Hieu Hoang and Philipp Koehn, 2009. A systematic analysis of translation model search spaces. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 224–232.

- Timothy C. Bell, J.G. Cleary and Ian H. Witten, 1990. Text compression: Prentice Hall, Englewood Cliffs, NJ.
- Ergun Biçici and Marc Dymetman. 2008. Dynamic translation memory: using statistical machine translation to improve translation memory fuzzy matches. In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2008)*, pages 454–465.
- Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. *Technical Report TR-10-98*, Harvard University Center for Research in Computing Technology.
- Yifan He, Yanjun Ma, Josef van Genabith and Andy Way, 2010. Bridging SMT and TM with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 622–630.
- Yifan He, Yanjun Ma, Andy Way and Josef van Genabith. 2011. Rich linguistic features for translation memory-inspired consistent translation. In *Proceedings of the Thirteenth Machine Translation Summit*, pages 456–463.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184.
- Katrin Kirchhoff, Jeff A. Bilmes and Kevin Duh. 2007. Factored language models tutorial. *Technical report*, Department of Electrical Engineering, University of Washington, Seattle, Washington, USA.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395, Barcelona, Spain.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer and Ondřej Bojar. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180.
- Philipp Koehn, Franz Josef Och and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54.
- Philipp Koehn and Jean Senellart. 2010. Convergence of translation memory and statistical machine translation. In *AMTA Workshop on MT Research and the Translation Industry*, pages 21–31.
- Elina Lagoudaki. 2006. Translation memories survey 2006: Users’ perceptions around tm use. In *Proceedings of the ASLIB International Conference Translating and the Computer 28*, pages 1–29.
- Qi Li, Biing-Hwang Juang, Qiru Zhou, and Chin-Hui Lee. 2000. Automatic verbal information verification for user authentication. *IEEE transactions on speech and audio processing*, Vol. 8, No. 5, pages 1063–6676.
- Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10 (8). pages 707–710.
- Yanjun Ma, Yifan He, Andy Way and Josef van Genabith. 2011. Consistent translation using discriminative learning: a translation memory-inspired approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 1239–1248, Portland, Oregon.
- Makoto Nagao, 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In: *Banerji, Alick Elithorn and Ranan (ed). Artificial and Human Intelligence: Edited Review Papers Presented at the International NATO Symposium on Artificial and Human Intelligence*. North-Holland, Amsterdam, 173–180.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29 (1). pages 19–51.
- Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.
- Aaron B. Phillips, 2011. Cunei: open-source machine translation with relevance-based models of each translation instance. *Machine Translation*, 25 (2). pages 166-177.
- Richard Sikes. 2007. Fuzzy matching in theory and practice. *Multilingual*, 18(6):39–43.
- Michel Simard and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *Proceedings of the Twelfth Machine Translation Summit (MT Summit XII)*, pages 120–127.
- James Smith and Stephen Clark. 2009. EBMT for SMT: a new EBMT-SMT hybrid. In *Proceedings of the 3rd International Workshop on Example-*

- Based Machine Translation (EBMT'09)*, pages 3–10, Dublin, Ireland.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA-2006)*, pages 223–231.
- Andreas Stolcke. 2002. SRILM—an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 311–318.
- Guillaume Wisniewski, Alexandre Allauzen and François Yvon, 2010. Assessing phrase-based translation models with oracle decoding. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 933–943.
- Ian H. Witten and Timothy C. Bell. 1991. The zero-frequency problem: estimating the probabilities of novel events in adaptive test compression. *IEEE Transactions on Information Theory*, 37(4): 1085–1094, July.
- Ventsislav Zhechev and Josef van Genabith. 2010. Seeding statistical machine translation with translation memory output through tree-based structural alignment. In *Proceedings of the 4th Workshop on Syntax and Structure in Statistical Translation*, pages 43–51.
- Andreas Zollmann, Ashish Venugopal, Franz Josef Och and Jay Ponte, 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1145–1152.

# Training Nondeficient Variants of IBM-3 and IBM-4 for Word Alignment

Thomas Schoenemann

Heinrich-Heine-Universität Düsseldorf, Germany

Universitätsstr. 1

40225 Düsseldorf, Germany

## Abstract

We derive variants of the fertility based models IBM-3 and IBM-4 that, while maintaining their zero and first order parameters, are nondeficient. Subsequently, we proceed to derive a method to compute a likely alignment and its neighbors as well as give a solution of EM training. The arising M-step energies are non-trivial and handled via projected gradient ascent.

Our evaluation on gold alignments shows substantial improvements (in weighted F-measure) for the IBM-3. For the IBM-4 there are no consistent improvements. Training the nondeficient IBM-5 in the regular way gives surprisingly good results.

Using the resulting alignments for phrase-based translation systems offers no clear insights w.r.t. BLEU scores.

## 1 Introduction

While most people think of the translation and word alignment models IBM-3 and IBM-4 as inherently deficient models (i.e. models that assign non-zero probability mass to impossible events), in this paper we derive nondeficient variants maintaining their zero order (IBM-3) and first order (IBM-4) parameters. This is possible as IBM-3 and IBM-4 are very special cases of general log-linear models: they are properly derived by the chain rule of probabilities. Deficiency is only introduced by ignoring a part of the history to be conditioned on in the individual factors of the chain rule factorization. While at first glance this seems necessary to obtain zero and first order de-

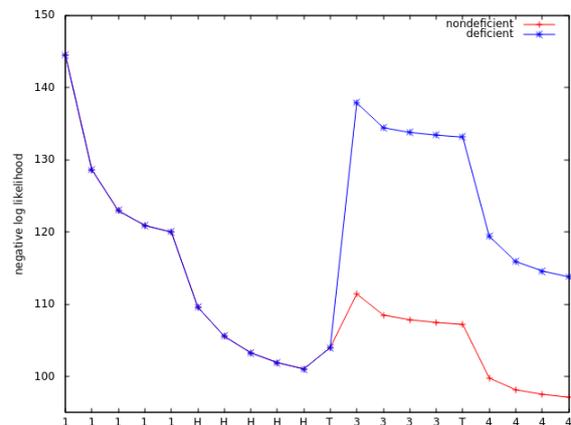


Figure 1: Plot of the negative log. likelihoods (the quantity to be minimized) arising in training deficient and nondeficient models (for Europarl German | English, training scheme  $1^5H^53^54^5$ ).  $1/3/4=IBM-1/3/4$ , H=HMM, T=Transfer iteration. The curves are identical up to iteration 11.

Iteration 11 shows that merely 5.14% of the (HMM) probability mass are covered by the Viterbi alignment and its neighbors. With deficient models (and deficient empty words) the final negative log likelihood is higher than the initial HMM one, with nondeficient models it is lower than for the HMM, as it should be for a better model.

dependencies, we show that with proper renormalization all factors can be made nondeficient.

Having introduced the model variants, we proceed to derive a hillclimbing method to compute a likely alignment (ideally the Viterbi alignment) and its neighbors. As for the deficient models, this plays an important role in the E-step of the subsequently derived expectation maximization (EM) training scheme. As usual, expectations in EM are approximated, but we now also get non-trivial M-step energies. We deal with these via projected gradient ascent.

The downside of our method is its resource consumption, but still we present results on corpora with 100.000 sentence pairs. The source code of this project is available in our word alignment software RegAligner<sup>1</sup>, version 1.2 and later.

Figure 1 gives a first demonstration of how much the proposed variants differ from the standard models by visualizing the resulting negative log likelihoods<sup>2</sup>, the quantity to be minimized in EM-training. The nondeficient IBM-4 derives a lower negative log likelihood than the HMM, the regular deficient variant only a lower one than the IBM-1. As an aside, the transfer iteration from HMM to IBM3 (iteration 11) reveals that only 5.14% of the probability mass<sup>3</sup> are preserved when using the Viterbi alignment and its neighbors instead of all alignments.

Indeed, it is widely recognized that – with proper initialization – fertility based models outperform sequence based ones. In particular, sequence based models can simply ignore a part of the sentence to be conditioned on, while fertility based models explicitly factor in a probability of words in this sentence to have no aligned words (or any other number of aligned words, called the *fertility*). Hence, it is encouraging to see that the nondeficient IBM-4 indeed derives a higher likelihood than the sequence based HMM.

**Related Work** Today’s most widely used models for word alignment are still the models IBM 1-5 of Brown et al. (1993) and the HMM of Vogel et al. (1996), thoroughly evaluated in (Och and Ney, 2003). While it is known that fertility-based models outperform sequence-based ones, the large bulk of word alignment literature following these publications has mostly ignored fertility-based models. This is different in the present paper which deals exclusively with such models.

One reason for the lack of interest is surely that computing expectations and Viterbi alignments for these models is a hard problem (Udupa and Maji, 2006). Nevertheless, computing Viterbi align-

ments for the IBM-3 has been shown to often be practicable (Ravi and Knight, 2010; Schoenemann, 2010).

Much work has been spent on HMM-based formulations, focusing on the computationally tractable side (Toutanova et al., 2002; Sumita et al., 2004; Deng and Byrne, 2005). In addition, some rather complex models have been proposed that usually aim to replace the fertility based models (Wang and Waibel, 1998; Fraser and Marcu, 2007a).

Another line of models (Melamed, 2000; Marcu and Wong, 2002; Cromières and Kurohashi, 2009) focuses on joint probabilities to get around the garbage collection effect (i.e. that for conditional models, rare words in the given language align to too many words in the predicted language). The downside is that these models are computationally harder to handle.

A more recent line of work introduces various forms of regularity terms, often in the form of symmetrization (Liang et al., 2006; Graça et al., 2010; Bansal et al., 2011) and recently by using  $L_0$  norms (Vaswani et al., 2012).

## 2 The models IBM-3, IBM-4 and IBM-5

We begin with a short review of fertility-based models in general and IBM-3, IBM-4 and IBM-5 specifically. All are due to (Brown et al., 1993) who proposed to use the deficient models IBM-3 and IBM-4 to initialize the nondeficient IBM-5.

For a foreign sentence  $\mathbf{f} = f_1^J = (f_1, \dots, f_J)$  with  $J$  words and an English one  $\mathbf{e} = e_1^I = (e_1, \dots, e_I)$  with  $I$  words, the (conditional) probability  $p(f_1^J | e_1^I)$  of getting the foreign sentence as a translation of the English one is modeled by introducing the word alignment  $\mathbf{a}$  as a hidden variable:

$$p(f_1^J | e_1^I) = \sum_{\mathbf{a}} p(f_1^J, \mathbf{a} | e_1^I)$$

All IBM models restrict the space of alignments to those where a foreign word can align to at most one target word. The resulting alignment is then written as a vector  $a_1^J$ , where each  $a_j$  takes integral values between 0 and  $I$ , with 0 indicating that  $f_j$  has no English correspondence.

The fertility-based models IBM-3, IBM-4 and IBM-5 factor the (conditional) probability  $p(f_1^J, a_1^J | e_1^I)$  of obtaining an alignment and a translation given an English sentence according to the following generative story:

<sup>1</sup><https://github.com/Thomas1205/RegAligner>, for the reported results we used a slightly earlier version.

<sup>2</sup>Note that the figure slightly favors IBM-1 and HMM as for them the length  $J$  of the foreign sequence is assumed to be known whereas IBM-3 and IBM-4 explicitly predict it.

<sup>3</sup>This number regards the corpus probability as in (9) to the power of  $1/S$ , i.e. the objective function in maximum likelihood training. The number is not entirely fair as alignments where more than half the words align to the empty word are assigned a probability of 0. Still, this is an issue only for short sentences.

1. For  $i = 1, 2, \dots, I$ , decide on the number  $\Phi_i$  of foreign words aligned to  $e_i$ . This number is called the *fertility* of  $e_i$ . Choose with probability  $p(\Phi_i|e_i^I, \Phi_1^{i-1}) = p(\Phi_i|e_i)$ .
2. Choose the number  $\Phi_0$  of unaligned words in the (still unknown) foreign sequence. Choose with probability  $p(\Phi_0|e_1^I, \Phi_1^I) = p(\Phi_0|\sum_{i=1}^I \Phi_i)$ . Since each foreign word belongs to exactly one English position (including 0), the foreign sequence is now known to be of length  $J = \sum_{i=0}^I \Phi_i$ .
3. For each  $i = 1, 2, \dots, I$ , and  $k = 1, \dots, \Phi_i$  decide on
  - (a) the identity  $f_{i,k}$  of the next foreign word aligned to  $e_i$ . Choose with probability  $p(f_{i,k}|e_i^I, \Phi_0^I, \mathbf{d}_1^{i-1}, d_{i,1}, \dots, d_{i,k-1}, \mathbf{f}_{i,k}) = p(f_{i,k}|e_i)$ , where  $\mathbf{d}_i$  comprises all  $d_{i,k}$  for word  $i$  (see point b) below) and  $\mathbf{f}_{i,k}$  comprises all foreign words known at that point.
  - (b) the position  $d_{i,k}$  of the just generated foreign word  $f_{i,k}$ , with probability  $p(d_{i,k}|e_i^I, \Phi_0^I, \mathbf{d}_1^{i-1}, d_{i,1}, \dots, d_{i,k-1}, \mathbf{f}_{i,k}, f_{i,k}) = p(d_{i,k}|e_i, \mathbf{d}_1^{i-1}, d_{i,1}, \dots, d_{i,k-1}, f_{i,k}, J)$ .
4. The remaining  $\Phi_0$  open positions in the foreign sequence align to position 0. Decide on the corresponding foreign words with  $p(f_{d_{0,k}}|e_0)$ , where  $e_0$  is an artificial ‘‘empty word’’.

To model the probability for the number of unaligned words in step 2, each of the  $\sum_{i=1}^I \Phi_i$  properly aligned foreign words generates an unaligned foreign word with probability  $p_0$ , resulting in

$$p(\Phi_0|\sum_{i=1}^I \Phi_i) = \binom{\sum_{i=1}^I \Phi_i}{\Phi_0} p_0^{\Phi_0} (1-p_0)^{(\sum_{i=1}^I \Phi_i) - \Phi_0},$$

with a base probability  $p_0$  and the combinatorial coefficients  $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ , where  $n! = \prod_{k=1}^n k$  denotes the factorial of  $n$ . The main difference between IBM-3, IBM-4 and IBM-5 is the choice of probability model in step 3 b), called a *distortion model*. The choices are now detailed.

## 2.1 IBM-3

The IBM-3 implements a zero order distortion model, resulting in

$$p(d_{i,k}|i, J) .$$

Since most of the context to be conditioned on is ignored, this allows invalid configurations to occur with non-zero probability: some foreign positions can be chosen several times, while others remain empty. One says that the model is *deficient*. On the other hand, the model for  $p(\Phi_0|\sum_{i=1}^I \Phi_i)$  is nondeficient, and in training this often results in very high probabilities  $p_0$ . To prevent this it is common to make this model deficient as well (Och and Ney, 2003), which improves performance immensely and gives much better results than simply fixing  $p_0$  in the original model.

As for each  $i$  the  $d_{i,k}$  can appear in any order (i.e. need not be in ascending order), there are  $\prod_{i=1}^I \Phi_i!$  ways to generate the same alignment  $a_1^J$  (where the  $\Phi_i$  are the fertilities induced by  $a_1^J$ ). In total, the IBM-3 has the following probability model:

$$p(f_1^J, a_1^J|e_1^I) = \prod_{j=1}^J \left[ p(f_j|e_{a_j}) \cdot p(j|a_j, J) \right] \cdot p(\Phi_0|\sum_{i=1}^I \Phi_i) \cdot \prod_{i=1}^I \Phi_i! p(\Phi_i|e_i) . \quad (1)$$

**Reducing the Number of Parameters** While using non-parametric models  $p(j|i, J)$  is convenient for closed-form M-steps in EM training, these parameters are not very intuitive. Instead, in this paper we use the *parametric* model

$$p(j|i, J) = \frac{p(j|i)}{\sum_{j=1}^J p(j|i)} \quad (2)$$

with the more intuitive parameters  $p(j|i)$ . The arising M-step energy is addressed by projected gradient ascent (see below).

These parameters are also used for the nondeficient variants. Using the original non-parametric ones can be handled in a very similar manner to the methods set forth below.

## 2.2 IBM-4

The distortion model of the IBM-4 is a first order one that generates the  $d_{i,k}$  of each English position  $i$  in ascending order (i.e. for  $1 < k \leq \Phi_i$  we have  $d_{i,k} > d_{i,k-1}$ ). There is then a one-to-one correspondence between alignments  $a_1^J$  and (valid) distortion parameters  $(d_{i,k})_{i=1, \dots, I, k=1, \dots, \Phi_i}$  and therefore no longer a factor of  $\prod_{i=1}^I \Phi_i!$ .

The IBM-4 has two sub-distortion models, one for the first aligned word ( $k = 1$ ) of an English position and one for all following words ( $k > 1$ , only

if  $\Phi_i > 1$ ). For position  $i$ , let  $[i] = \arg \max\{i' | 1 \leq i' < i, \Phi_{i'} > 0\}$  denote<sup>4</sup> the closest preceding English word that has aligned foreign words. The aligned foreign positions of  $[i]$  are combined into a *center position*  $\odot_{[i]}$ , the rounded average of the positions. Now, the distortion probability for the first word ( $k = 1$ ) is

$$p_{=1}(d_{i,1} | \odot_{[i]}, \mathcal{A}(f_{i,1}), \mathcal{B}(e_{[i]}), J),$$

where  $\mathcal{A}$  gives the word class of a foreign word and  $\mathcal{B}$  the word class of an English word (there are typically 50 classes per language, derived by machine learning techniques). The probability is further reduced to a dependency on the difference of the positions, i.e.  $p_{=1}(d_{i,1} - \odot_{[i]} | \mathcal{A}(f_{i,1}), \mathcal{B}(e_{[i]}))$ . For  $k > 1$  the model is

$$p_{>1}(d_{i,k} | d_{i,k-1}, \mathcal{A}(f_{i,k}), J),$$

which is likewise reduced to  $p_{>1}(d_{i,k} - d_{i,k-1} | \mathcal{A}(f_{i,k}))$ . Note that in both difference-based formulations the dependence on  $J$  has to be dropped to get closed-form solutions of the M-step in EM training, and Brown et al. note themselves that the IBM-4 can place words before the start and after the end of the sentence.

**Reducing Deficiency** In this paper, we also investigate the effect of reducing the amount of wasted probability mass by enforcing the dependence on  $J$  by proper renormalization, i.e. using

$$p_{=1}(j | j', \mathcal{A}(f_{i,1}), \mathcal{B}(e_{[i]}), J) = \frac{p_{=1}(j - j' | \mathcal{A}(f_{i,1}), \mathcal{B}(e_{[i]}))}{\sum_{j''=1}^J p_{=1}(j'' - j' | \mathcal{A}(f_{i,1}), \mathcal{B}(e_{[i]}))}, \quad (3)$$

for the first aligned word and

$$p_{>1}(j | j', \mathcal{A}(f_{i,k}), J) = \frac{p_{>1}(j - j' | \mathcal{A}(f_{i,k}))}{\sum_{j''=1}^J p_{>1}(j'' - j' | \mathcal{A}(f_{i,k}))} \quad (4)$$

for all following words, again handling the M-step in EM training via projected gradient ascent. With this strategy words can no longer be placed outside the sentence, but a lot of probability mass is still wasted on configurations where at least one foreign (or predicted) position  $j$  aligns to two or more positions  $i, i'$  in the English (or given) language (and consequently there are more unaligned

<sup>4</sup>If the set is empty, instead a sentence start probability is used. Note that we differ slightly in notation compared to (Brown et al., 1993).

source words than the generated  $\Phi_0$ ). Therefore, here, too, the probability for  $\Phi_0$  has to be made deficient to get good performance.

In summary, the base model for the IBM-4 is:

$$p(f_1^J, a_1^J | e_1^I) = p\left(\Phi_0 \mid \sum_{i=1}^I \Phi_i\right) \cdot \prod_{j=1}^J p(f_j | e_{a_j}) \cdot \prod_{i=1}^I p(\Phi_i | e_i) \cdot \prod_{i:\Phi_i>0} \left[ p_{=1}(d_{i,1} - \odot_{[i]} | \mathcal{A}(f_{i,1}), \mathcal{B}(e_{[i]})) \cdot \prod_{k=2}^{\Phi_i} p_{>1}(d_{i,k} - d_{i,k-1} | \mathcal{A}(f_{i,k})) \right], \quad (5)$$

where empty products are understood to be 1.

### 2.3 IBM-5

We note in passing that the distortion model of the IBM-5 is nondeficient and has parameters for filling the  $n$ th open gap in the foreign sequence given that there are  $N$  positions to choose from – see the next section for exactly what positions one can choose from. There is also a dependence on word classes for the foreign language.

This is neither a zero order nor a first order dependence, and in (Och and Ney, 2003) the first order model of the IBM-4, though deficient, outperformed the IBM-5. The IBM-5 is therefore rarely used in practice. This motivated us to instead reformulate IBM-3 and IBM-4 as nondeficient models. In our results, however, the IBM-5 gave surprisingly good results and was often superior to all variants of the IBM-4.

## 3 Nondeficient Variants of IBM-3 and IBM-4

From now on we always enforce that for each position  $i$  the indices  $d_{i,k}$  are generated in ascending order ( $d_{i,k} > d_{i,k-1}$  for  $k > 1$ ). A central concept for the generation of  $d_{i,k}$  in step 3(b) is the set of positions in the foreign sequence that are still without alignment. We denote the set of these positions by

$$\mathcal{J}_{i,k,J} = \{1, \dots, J\} - \{d_{i,k'} | 1 \leq k' < k\} - \{d_{i',k'} | 1 \leq i' < i, 1 \leq k' \leq \Phi_{i'}\}$$

where the dependence on the various  $d_{i',k'}$  is not made explicit in the following.

It is tempting to think that in a nondeficient model all members of  $\mathcal{J}_{i,k,J}$  can be chosen for

$d_{i,k}$ , but this holds only  $\Phi_i = 1$ . Otherwise, the requirement of generating the  $d_{i,k}$  in ascending order prevents us from choosing the  $(\Phi_i - k)$  largest entries in  $\mathcal{J}_{i,k,J}$ . For  $k > 1$  we also have to remove all positions smaller than  $d_{i,k-1}$ .

Let  $\mathcal{J}_{i,k,J}^{\Phi_i}$  denote the set where these positions have been removed. With that, we can state the nondeficient variants of IBM-3 and IBM-4.

### 3.1 Nondeficient IBM-3

For the IBM-3, we define the auxiliary quantity

$$q(d_{i,k} = j | i, \mathcal{J}_{i,k,J}^{\Phi_i}) = \begin{cases} p(j|i) & \text{if } j \in \mathcal{J}_{i,k,J}^{\Phi_i} \\ 0 & \text{else,} \end{cases}$$

where we use the zero order parameters  $p(j|i)$  we also use for the standard (deficient) IBM-3, compare (2). To get a nondeficient variant, it remains to renormalize, resulting in

$$p(d_{i,k} = j | i, \mathcal{J}_{i,k,J}^{\Phi_i}) = \frac{q(j|i, \mathcal{J}_{i,k,J}^{\Phi_i})}{\sum_{j=1}^J q(j|i, \mathcal{J}_{i,k,J}^{\Phi_i})}. \quad (6)$$

Further, note that the factors  $\Phi_i!$  now have to be removed from (1) as the  $d_{i,k}$  are generated in ascending order. Lastly, here we use the original nondeficient empty word model  $p(\Phi_0 | \sum_{i=1}^I \Phi_i)$ , resulting in a totally nondeficient model.

### 3.2 Nondeficient IBM-4

With the notation set up, it is rather straightforward to derive a nondeficient variant of the IBM-4. Here, there are the two cases  $k = 1$  and  $k > 1$ . We begin with the case  $k = 1$ . Abbreviating  $\alpha = \mathcal{A}(f_{i,1})$  and  $\beta = \mathcal{B}(e_{[i]})$ , we define the auxiliary quantity

$$q_{=1}(d_{i,1} = j | \odot_{[i]}, \alpha, \beta, \mathcal{J}_{i,k,J}^{\Phi_i}) = \begin{cases} p_{=1}(j - \odot_{[i]} | \alpha, \beta) & \text{if } j \in \mathcal{J}_{i,k,J}^{\Phi_i} \\ 0 & \text{else,} \end{cases} \quad (7)$$

again using the - now first order - parameters of the base model. The nondeficient distribution  $p_{=1}(d_{i,1} = j | \odot_{[i]}, \alpha, \beta, \mathcal{J}_{i,k,J}^{\Phi_i})$  is again obtained by renormalization.

For the case  $k > 1$ , we abbreviate  $\alpha = \mathcal{A}(f_{i,k})$  and introduce the auxiliary quantity

$$q_{>1}(d_{i,k} = j | d_{i,k-1}, \alpha, \mathcal{J}_{i,k,J}^{\Phi_i}) = \begin{cases} p_{>1}(j - d_{i,k-1} | \alpha) & \text{if } j \in \mathcal{J}_{i,k,J}^{\Phi_i} \\ 0 & \text{else,} \end{cases} \quad (8)$$

from which the nondeficient distribution  $p_{>1}(d_{i,k} = j | d_{i,k-1}, \alpha, \mathcal{J}_{i,k,J}^{\Phi_i})$  is again obtained by renormalization.

## 4 Training the New Variants

For the task of word alignment, we infer the parameters of the models using the maximum likelihood criterion

$$\max_{\theta} \prod_{s=1}^S p_{\theta}(\mathbf{f}_s | \mathbf{e}_s) \quad (9)$$

on a set of training data (i.e. sentence pairs  $s = 1, \dots, S$ ). Here,  $\theta$  comprises all base parameters of the respective model (e.g. for the IBM-3 all  $p(f|e)$ , all  $p(\Phi, e)$  and all  $p(j|i)$ ) and  $p_{\theta}$  signifies the dependence of the model on the parameters. Note that (9) is truly a *constrained* optimization problem as the parameters  $\theta$  have to satisfy a number of probability normalization constraints.

When  $p_{\theta}(\cdot)$  denotes a fertility based model the resulting problem is a non-concave maximization problem with many local minima and no (known) closed-form solutions. Hence, it is handled by computational methods, which typically apply the logarithm to the above function.

Our method of choice to attack the maximum likelihood problem is expectation maximization (EM), the standard in the field, which we explain below. Due to non-concaveness the starting point for EM is of extreme importance. As is common, we first train an IBM-1 and then an HMM before proceeding to the IBM-3 and finally the IBM-4.

As in the training of the deficient IBM-3 and IBM-4 models, we approximate the expectations in the E-step by a set of likely alignments, ideally centered around the Viterbi alignment, but already for the regular deficient variants computing it is NP-hard (Udapa and Maji, 2006). A first task is therefore to compute such a set. This task is also needed for the actual task of word alignment (annotating a given sentence pair with an alignment).

### 4.1 Alignment Computation

For computing alignments, we use the common procedure of hillclimbing where we start with an alignment, then iteratively compute the probabilities of all alignments differing by a move or a swap (Brown et al., 1993) and move to the best of these if it beats the current alignment.

Since we cannot ignore parts of the history and still get a nondeficient model, computing the probabilities of the neighbors cannot be handled incrementally (or rather only partially, for the dictionary and fertility models). While this does increase running times, in practice the M-steps take longer than the E-steps.

For self-containment, we recall here that for an alignment  $a_1^J$  applying the **move**  $a_1^J[j \rightarrow i]$  results in the alignment  $\hat{a}_1^J$  defined by  $\hat{a}_j = i$  and  $\hat{a}_{j'} = a_{j'}$  for  $j' \neq j$ . Applying the **swap**  $a_1^J[j_1 \leftrightarrow j_2]$  results in the alignment  $\hat{a}_1^J$  defined by  $\hat{a}_{j_1} = a_{j_2}$ ,  $\hat{a}_{j_2} = a_{j_1}$  and  $\hat{a}_{j'} = a_{j'}$  elsewhere. If  $a_1^J$  is the alignment produced by hillclimbing, the **move matrix**  $m \in \mathbb{R}^{J \times I+1}$  is defined by  $m_{j,i}$  being the probability of  $a_1^J[j \rightarrow i]$  as long as  $a_j \neq i$ , otherwise 0. Likewise the **swap matrix**  $s \in \mathbb{R}^{J \times J}$  is defined as  $s_{j_1, j_2}$  being the probability of  $a_1^J[j_1 \leftrightarrow j_2]$  for  $a_{j_1} \neq a_{j_2}$ , 0 otherwise. The move and swap matrices are used to approximate expectations in EM training (see below).

## 4.2 Parameter Update

**Naive Scheme** It is tempting to account for the changes in the model in hillclimbing, but to otherwise use the regular M-step procedures (closed form solution when not conditioning on  $J$  for the IBM-4 and for the non-parametric IBM-3, otherwise projected gradient ascent) for the deficient models. However, we verified that this is not a good idea: not only can the likelihood go down in the process (even if we could compute expectations exactly), but these schemes also heavily increase  $p_0$  in each iteration, i.e. the same problem Och and Ney (2003) found for the deficient models. There is therefore the need to execute the M-step properly, and when done the problem is indeed resolved.

**Proper EM** The expectation maximization (EM) framework (Dempster et al., 1977; Neal and Hinton, 1998) is a class of template procedures (rather than a proper algorithm) that iteratively requires solving the task

$$\max_{\theta_k} \sum_{s=1}^S \sum_{\mathbf{a}_s} p_{\theta_{k-1}}(\mathbf{a}_s | \mathbf{f}_s, \mathbf{e}_s) \log(p_{\theta_k}(\mathbf{f}_s, \mathbf{a}_s | \mathbf{e}_s)) \quad (10)$$

by appropriate means. Here,  $\theta_{k-1}$  are the parameters from the previous iteration, while  $\theta_k$  are those derived in the current iteration. Of course, here and in the following the normalization constraints on  $\theta$  apply, as already in (9). On explicit request of a reviewer we give a detailed account for our setting here. Readers not interested in the details can safely move on to the next section.

**Details on EM** For the corpora occurring in practice, the function (10) has many more terms than there are atoms in the universe. The trick is

that  $p_{\theta_k}(\mathbf{f}_s, \mathbf{a}_s | \mathbf{e}_s)$  is a product of factors, where each factor depends on very few components of  $\theta_k$  only. Taking the logarithm gives a sum of logarithms, and in the end we are left with the problem of computing the weights of each factor, which turn out to be expectations. To apply this to the (deficient) **IBM-3** model with parametric distortion we simplify  $p_{\theta_{k-1}}(\mathbf{a}_s | \mathbf{f}_s, \mathbf{e}_s) = p(\mathbf{a}_s)$  and define the counts  $n_{f,e}(\mathbf{a}_s) = \sum_{j=1}^J \delta(f_j^s, f) \cdot \delta(e_{a_j^s}^s, e)$ ,  $n_{\Phi,e}(\mathbf{a}_s) = \sum_{i=1}^I \delta(e_i^s, e) \cdot \delta(\Phi_i(\mathbf{a}_s), \Phi)$  and  $n_{j,i}(\mathbf{a}_s) = \delta(a_{j_1}^s, i)$ . We also use short hand notations for sets, e.g.  $\{p(f|e)\}$  is meant as the set of all translation probabilities induced by the given corpus. With this notation, after reordering the terms problem (10) can be written as

$$\begin{aligned} & \max_{\{p(f|e)\}, \{p(\Phi|e)\}, \{p(j|i)\}} \quad (11) \\ & \sum_{e,f} \left[ \sum_{s=1}^S \sum_{\mathbf{a}_s} p(\mathbf{a}_s) n_{f,e}(\mathbf{a}_s) \right] \log(p(f|e)) \\ & + \sum_{e,\Phi} \left[ \sum_{s=1}^S \sum_{\mathbf{a}_s} p(\mathbf{a}_s) n_{\Phi,e}(\mathbf{a}_s) \right] \log(p(\Phi|e)) \\ & + \sum_{i,j} \left[ \sum_{s=1}^S \sum_{\mathbf{a}_s} p(\mathbf{a}_s) n_{j,i}(\mathbf{a}_s) \right] \log(p(j|i, J)). \end{aligned}$$

Indeed, the weights in each line turn out to be nothing else than expectations of the respective factor under the distribution  $p_{\theta_{k-1}}(\mathbf{a}_s | \mathbf{f}_s, \mathbf{e}_s)$  and will henceforth be written as  $w_{f,e}$ ,  $w_{\Phi,e}$  and  $w_{j,i,J}$ . Therefore, executing an iteration of EM requires first calculating all expectations (E-step) and then solving the maximization problems (M-step). For models such as IBM-1 and HMM the expectations can be calculated efficiently, so the enormous sum of terms in (10) is equivalently written as a manageable one. In this case it can be shown<sup>5</sup> that the new  $\theta_k$  must have a higher likelihood (9) than  $\theta_{k-1}$  (unless a stationary point is reached). In fact, any  $\theta$  that has a higher value in the auxiliary function (11) than  $\theta_{k-1}$  must also have a higher likelihood. This is an important background for parametric models such as (2) where the M-step cannot be solved exactly.

For IBM-3/4/5 computing exact expectations is intractable (Udapa and Maji, 2006) and approximations have to be used (in fact, even computing the likelihood for a given  $\theta$  is intractable). We

<sup>5</sup>See e.g. the author's course notes (in German), currently <http://user.phil-fak.uni-duesseldorf.de/~tosch/downloads/statmt/wordalign.pdf>.

use the common procedure based on hillclimbing and the move/swap matrices. The likelihood is not guaranteed to increase but it (or rather its approximation) always did in each of the five run iterations. Nevertheless, the main advantage of EM is preserved: problem (11) decomposes into several smaller problems, one for each probability distribution since the parameters are tied by the normalization constraints. The result is one problem for each  $e$  involving all  $p(f|e)$ , one for each  $e$  involving all  $p(\Phi|e)$  and one for each  $i$  involving all  $p(j|i)$ .

The problems for the translation probabilities and the fertility probabilities yield the known standard update rules. The most interesting case is the problem for the (parametric) distortion models. In the deficient setting, the problem for each  $i$  is

$$\max_{\{p(j|i)\}} \sum_J w_{i,j,J} \log \left( \frac{p(j|i)}{\sum_{j'=1}^J p(j'|i)} \right)$$

In the nondeficient setting, we now drop the subscripts  $i, k, J$  and the superscript  $\Phi$  from the sets defined in the previous sections, i.e. we write  $\mathcal{J}$  instead of  $\mathcal{J}_{i,k,J}^\Phi$ . The M-step problem is then

$$\max_{\{p(j|i)\}} E_i = \sum_j \sum_{\mathcal{J}:j \in \mathcal{J}} w_{j,i,\mathcal{J}} \log(p(j|i, \mathcal{J})),$$

where  $w_{j,i,\mathcal{J}}$  (with  $j \in \mathcal{J}$ ) is the expectation for aligning  $j$  to  $i$  when one can choose among the positions in  $\mathcal{J}$ , and with  $p(j|i, \mathcal{J})$  as in (6). In principle there is an exponential number of expectations  $w_{j,i,\mathcal{J}}$ . However, since we approximate expectations from the move and swap matrices, and hence by  $\mathcal{O}((I+J) \cdot J)$  alignments per sentence pair, in the end we get a polynomial number of terms. Currently we only consider alignments with (approximated)  $p_{\theta_{k-1}}(\mathbf{a}_s | \mathbf{f}_s, \mathbf{e}_s) > 10^{-6}$ .

Importantly, the fact that we get separate M-step problems for different  $i$  allows us to reduce memory consumption by using refined data structures when storing the expectations.

For both the deficient and the nondeficient variants, the M-step problems for the distortion parameters  $p(j|i)$  are non-trivial, non-concave and have no (known) closed form solutions. We approach them via the method of projected gradient ascent (PGA), where the gradient for the nondeficient problem is

$$\frac{\partial E_i}{\partial p(j|i)} = \sum_{\mathcal{J}:j \in \mathcal{J}} \left[ \frac{w_{j,\mathcal{J}}}{p(j|i)} - \frac{\sum_{j' \in \mathcal{J}} w_{j',\mathcal{J}}}{\sum_{j' \in \mathcal{J}} p(j'|i)} \right].$$

When running PGA we guarantee that the resulting  $\{p(j|i)\}$  has a higher function value  $E_i$  than the input ones (unless a stationary point is input). We stop when a cutoff criterion indicates a local maximum or 250 iterations are used up.

**Projected Gradient Ascent** This method is used in a couple of recent papers, notably (Schoenemann, 2011; Vaswani et al., 2012) and is briefly sketched here for self-containment (see those papers for more details). To solve a maximization problem

$$\max_{p(j|i) \geq 0, \sum_j p(j|i)=1} E_i(\{p(j|i)\})$$

for some (differentiable) function  $E_i(\cdot)$ , one iteratively starts at the current point  $\{p_k(j|i)\}$ , computes the gradient  $\nabla E_i(\{p_k(j|i)\})$  and goes to the point

$$q(j|i) = p_k(j|i) + \alpha \nabla E_i(p_k(j|i)), j = 1, \dots, J$$

for some step-length  $\alpha$ . This point is generally not a probability distribution, so one computes the *nearest* probability distribution

$$\min_{q'(j|i) \geq 0, \sum_j q'(j|i)=1} \sum_{j=1}^J (q'(j|i) - q(j|i))^2,$$

a step known as *projection* which we solve with the method of (Michelot, 1986). The new distribution  $\{q'(j|i)\}$  is not guaranteed to have a higher  $E_i(\cdot)$ , but (since the constraint set is a convex one) a suitable interpolation of  $\{p_k(j|i)\}$  and  $\{q'(j|i)\}$  is guaranteed to have a higher value (unless  $\{p_k(j|i)\}$  is a local maximum or minimum of  $E_i(\cdot)$ ). Such a point is computed by backtracking line search and defines the next iterate  $\{p_{k+1}(j|i)\}$ .

**IBM-4** When moving from the IBM-3 to the IBM-4, only the last line in (11) changes. In the end one gets two new kinds of problems, for  $p_{=1}(\cdot)$  and  $p_{>1}(\cdot)$ . For  $p_{=1}(\cdot)$  we have one problem for each foreign class  $\alpha$  and each English class  $\beta$ , of the form

$$\max_{\{p_{=1}(j|j',\alpha,\beta)\}} \sum_{j,j',J} w_{j,j',J,\alpha,\beta} \log(p_{=1}(j|j',\alpha,\beta, J))$$

for reduced deficiency (with  $p_{=1}(j|j',\alpha,\beta, J)$  as in (3)) and of the form

$$\max_{\{p_{>1}(j|j',\alpha,\beta)\}} \sum_{j,j',J} w_{j,j',J,\alpha,\beta} \log(p_{>1}(j|j',\alpha,\beta, J))$$

Model	Degree of Deficiency	De En	En De	Es En	En Es
HMM	nondeficient (our)	73.8	77.6	77.4	76.1
IBM-3	full (GIZA++)	74.2	76.5	74.3	74.5
IBM-3	full (our)	75.6	79.2	75.2	73.7
IBM-3	nondeficient (our)	76.1	79.8	76.8	75.5
IBM-4, 1 x 1 word class	full (GIZA++)	77.9	79.4	78.6	78.4
IBM-4, 1 x 1 word class	full (our)	76.1	81.5	77.8	78.0
IBM-4, 1 x 1 word class	reduced (our)	77.2	80.6	77.9	78.3
IBM-4, 1 x 1 word class	nondeficient (our)	77.6	81.5	80.0	78.4
IBM-4, 50 x 50 word classes	full (GIZA++)	78.6	80.4	79.3	79.3
IBM-4, 50 x 50 word classes	full (our)	78.0	82.4	79.2	79.4
IBM-4, 50 x 50 word classes	reduced (our)	78.5	82.1	79.2	79.0
IBM-4, 50 x 50 word classes	nondeficient (our)	77.9	82.5	79.7	78.2
IBM-5, 50 word classes	nondeficient (GIZA++)	79.4	81.1	80.0	79.5
IBM-5, 50 word classes	nondeficient (our)	79.2	82.7	79.7	79.5

Table 1: **Alignment accuracy** (weighted F-measure times 100,  $\alpha = 0.1$ ) on Europarl with 100.000 sentence pairs. Reduced deficiency means renormalization as in (3) and (4), so that words cannot be placed before or after the sentence. **For the IBM-3**, the nondeficient variant is clearly best. **For the IBM-4** it is better in roughly half the cases, both with and without word classes.

for the nondeficient variant, with  $p_{=1}(j|j', \alpha, \beta, \mathcal{J})$  based on (7).

For  $p_{>1}(\cdot)$  we have one problem per foreign class  $\alpha$ , of the form

$$\max_{\{p_{>1}(j|j', \alpha)\}} \sum_{j, j', J} w_{j, j', J, \alpha} \log(p_{>1}(j|j', \alpha, J))$$

for reduced deficiency, with  $p_{>1}(j|j', \alpha, J)$  based on (4), and for the nondeficient variant it has the form

$$\max_{\{p_{>1}(j|j', \alpha)\}} \sum_{j, j', \mathcal{J}} w_{j, j', \mathcal{J}, \alpha} \log(p_{>1}(j|j', \alpha, \mathcal{J})),$$

with  $p_{>1}(j|j', \alpha, \mathcal{J})$  based on (8). Calculating the gradients is analogous to the IBM-3.

## 5 Experiments

We test the proposed methods on subsets of the Europarl corpus for German and English as well as Spanish and English, using lower-cased corpora. We evaluate alignment accuracies on gold alignments<sup>6</sup> in the form of weighted F-measures with  $\alpha = 0.1$ , which performed well in (Fraser and Marcu, 2007b). In addition we evaluate the effect on phrase-based translation on one of the tasks.

We implement the proposed methods in our own framework RegAligner rather than GIZA++,

<sup>6</sup>from (Lambert et al., 2005) and from <http://user.phil-fak.uni-duesseldorf.de/~tosch/downloads.html>.

which is only rudimentally maintained. Therefore, we compare to the deficient models in our own software as well as to those in GIZA++.

We run 5 iterations of IBM-1, followed by 5 iterations of HMM, 5 of IBM-3 and finally 5 of IBM-4. The first iteration of the IBM-3 collects counts from the HMM, and likewise the first iteration of the IBM-4 collects counts from the IBM-3 (in both cases the move and swap matrices are filled with probabilities of the former model, then these matrices are used as in a regular model iteration). A nondeficient IBM-4 is always initialized by a nondeficient IBM-3. We did not set a fertility limit (except for GIZA++).

Experiments were run on a Core i5 with 2.5 GHz and 8 GB of memory. The latter was the main reason why we did not use still larger corpora<sup>7</sup>. The running times for the entire training were half a day without word classes and a day with word classes. With 50 instead of 250 PGA iterations in all M-steps we get only half these running times, but the resulting F-measures deteriorate, especially for the IBM-4 with classes.

The running times of our implementation of the IBM-5 are much more favorable: the entire training then runs in little more than an hour.

<sup>7</sup>The main memory bottleneck is the IBM-4 (6 GB without classes, 8 GB with). Using refined data structures should reduce this bottleneck.

## 5.1 Alignment Accuracy

The alignment accuracies – weighted F-measures with  $\alpha = 0.1$  – for the tested corpora and model variants are given in Table 1. Clearly, nondeficiency greatly improves the accuracy of the IBM-3, both compared to our deficient implementation and that of GIZA++.

For the IBM-4 we get improvements for the nondeficient variant in roughly half the cases, both with and without word classes. We think this is an issue of local minima, inexactly solved M-steps and sensitiveness to initialization.

Interestingly, also the reduced deficient IBM-4 is not always better than the fully deficient variant. Again, we think this is due to problems with the non-concave nature of the models.

There is also quite some surprise regarding the IBM-5: contrary to the findings of (Och and Ney, 2003) the IBM-5 in GIZA++ performs best in three out of four cases - when competing with both deficient and nondeficient variants of IBM-3 and IBM-4. Our own implementation gives slightly different results (as we do not use smoothing), but it, too, performs very well.

## 5.2 Effect on Translation Performance

We also check the effect of the various alignments (all produced by RegAligner) on translation performance for phrase-based translation, randomly choosing translation from German to English. We use MOSES with a 5-gram language model (trained on 500.000 sentence pairs) and the standard setup in the MOSES Experiment Management System: training is run in both directions, the alignments are combined using `diag-grow-final-and` (Och and Ney, 2003) and the parameters of MOSES are optimized on 750 development sentences.

The resulting BLEU-scores are shown in Table 2. However, the table shows no clear trends and even the IBM-3 is not clearly inferior to the IBM-4. We think that one would need to handle larger corpora (or run multiple instances of Minimum Error Rate Training with different random seeds) to get more meaningful insights. Hence, at present our paper is primarily of theoretical value.

## 6 Conclusion

We have shown that the word alignment models IBM-3 and IBM-4 can be turned into nondeficient

Model	#Classes	Deficiency	BLEU
HMM	-	nondeficient	29.72
IBM-3	-	deficient	29.63
IBM-3	-	nondeficient	29.73
IBM-4	1 x 1	fully deficient	29.91
IBM-4	1 x 1	reduced deficient	29.88
IBM-4	1 x 1	nondeficient	30.18
IBM-4	50 x 50	fully deficient	29.86
IBM-4	50 x 50	reduced deficient	30.14
IBM-4	50 x 50	nondeficient	29.90
IBM-5	50	nondeficient	29.84

Table 2: Evaluation of **phrase-based translation** from German to English with the obtained alignments (for 100.000 sentence pairs). Training is run in both directions and the resulting alignments are combined via `diag-grow-final-and`. The table shows no clear superiority of any method. In fact, the IBM-4 is not superior to the IBM-3 and the HMM is about equal to the IBM-3. We think that one needs to handle larger corpora to get clearer insights.

variants, an important aim of probabilistic modeling for word alignment.

Here we have exploited that the models are proper applications of the chain rule of probabilities, where deficiency is only introduced by ignoring parts of the history for the distortion factors in the factorization. By proper renormalization the desired nondeficient variants are obtained.

The arising models are trained via expectation maximization. In the E-step we use hillclimbing to get a likely alignment (ideally the Viterbi alignment). While this cannot be handled fully incrementally, it is still fast enough in practice. The M-step energies are non-concave and have no (known) closed-form solutions. They are handled via projected gradient ascent.

For the IBM-3 nondeficiency clearly improves alignment accuracy. For the IBM-4 we get improved accuracies in roughly half the cases, both with and without word classes. The IBM-5 performs surprisingly well, it is often best and hence much better than its reputation. An evaluation of phrase based translation showed no clear insights.

Nevertheless, we think that nondeficiency in fertility based models is an important issue, and that at the very least our paper is of theoretical value. The implementations are publicly available in RegAligner 1.2.

## References

- M. Bansal, C. Quirk, and R. Moore. 2011. Gappy phrasal alignment by agreement. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, Portland, Oregon, June.
- P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, and R.L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, June.
- F. Cromières and S. Kurohashi. 2009. An alignment algorithm using Belief Propagation and a structure-based distortion model. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Athens, Greece, April.
- A.P. Dempster, N.M. Laird, and D.B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.
- Y. Deng and W. Byrne. 2005. HMM word and phrase alignment for statistical machine translation. In *HLT-EMNLP*, Vancouver, Canada, October.
- A. Fraser and D. Marcu. 2007a. Getting the structure right for word alignment: LEAF. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Prague, Czech Republic, June.
- A. Fraser and D. Marcu. 2007b. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303, September.
- J. Graça, K. Ganchev, and B. Taskar. 2010. Learning tractable word alignment models with complex constraints. *Computational Linguistics*, 36, September.
- P. Lambert, A.D. Gispert, R. Banchs, and J.B. Marino. 2005. Guidelines for word alignment evaluation and manual alignment. *Language Resources and Evaluation*, 39(4):267–285.
- P. Liang, B. Taskar, and D. Klein. 2006. Alignment by agreement. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, New York, New York, June.
- D. Marcu and W. Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, Pennsylvania, July.
- D. Melamed. 2000. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–249.
- C. Michelot. 1986. A finite algorithm for finding the projection of a point onto the canonical simplex of  $\mathbb{R}^n$ . *Journal on Optimization Theory and Applications*, 50(1), July.
- R.M. Neal and G.E. Hinton. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M.I. Jordan, editor, *Learning in Graphical Models*. MIT press.
- F.J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- S. Ravi and K. Knight. 2010. Does GIZA++ make search errors? *Computational Linguistics*, 36(3).
- T. Schoenemann. 2010. Computing optimal alignments for the IBM-3 translation model. In *Conference on Computational Natural Language Learning (CoNLL)*, Uppsala, Sweden, July.
- T. Schoenemann. 2011. Regularizing mono- and bi-word models for word alignment. In *International Joint Conference on Natural Language Processing (IJCNLP)*, Chiang Mai, Thailand, November.
- E. Sumita, Y. Akiba, T. Doi, A. Finch, K. Imamura, H. Okuma, M. Paul, M. Shimohata, and T. Watanabe. 2004. EBMT, SMT, Hybrid and more: ATR spoken language translation system. In *International Workshop on Spoken Language Translation (IWSLT)*, Kyoto, Japan, September.
- K. Toutanova, H.T. Ilhan, and C.D. Manning. 2002. Extensions to HMM-based statistical word alignment models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, Pennsylvania, July.
- R. Udupa and H.K. Maji. 2006. Computational complexity of statistical machine translation. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Trento, Italy, April.
- A. Vaswani, L. Huang, and D. Chiang. 2012. Smaller alignment models for better translations: Unsupervised word alignment with the  $l_0$ -norm. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, Jeju, Korea, July.
- S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *International Conference on Computational Linguistics (COLING)*, pages 836–841, Copenhagen, Denmark, August.
- Y.-Y. Wang and A. Waibel. 1998. Modeling with structures in statistical machine translation. In *International Conference on Computational Linguistics (COLING)*, Montreal, Canada, August.

# Modelling Annotator Bias with Multi-task Gaussian Processes: An Application to Machine Translation Quality Estimation

Trevor Cohn and Lucia Specia

Department of Computer Science

University of Sheffield

Sheffield, United Kingdom

{t.cohn, l.specia}@sheffield.ac.uk

## Abstract

Annotating linguistic data is often a complex, time consuming and expensive endeavour. Even with strict annotation guidelines, human subjects often deviate in their analyses, each bringing different biases, interpretations of the task and levels of consistency. We present novel techniques for learning from the outputs of multiple annotators while accounting for annotator specific behaviour. These techniques use multi-task Gaussian Processes to learn jointly a series of annotator and metadata specific models, while explicitly representing correlations between models which can be learned directly from data. Our experiments on two machine translation quality estimation datasets show uniform significant accuracy gains from multi-task learning, and consistently outperform strong baselines.

## 1 Introduction

Most empirical work in Natural Language Processing (NLP) is based on supervised machine learning techniques which rely on human annotated data of some form or another. The annotation process is often time consuming, expensive, and prone to errors; moreover there is often considerable disagreement amongst annotators.

In general, the predominant perspective to deal with these data annotation issues in previous work has been that there is a single underlying *ground truth*, and that the annotations collected are noisy and/or biased samples of this. The challenge is then one of quality control, in order to process the data by filtering, averaging or similar to distil the truth. We posit that this perspective is too limiting, especially with respect to linguistic data, where each individual's idiolect and linguistic background can give rise to many different

– and yet equally valid – truths. Particularly in highly subjective annotation tasks, the differences between annotators cannot be captured by simple models such as scaling all instances of a certain annotator by a factor. They can originate from a number of nuanced aspects. This is the case, for example, of annotations on the quality of sentences generated using machine translation (MT) systems, which are often used to build **quality estimation** models (Blatz et al., 2004; Specia et al., 2009) – our application of interest.

In addition to annotators' own perceptions and expectations with respect to translation quality, a number of factors can affect their judgements on specific sentences. For example, certain annotators may prefer translations produced by rule-based systems as these tend to be more grammatical, while others would prefer sentences produced by statistical systems with more adequate lexical choices. Likewise, some annotators can be biased by the complexity of the source sentence: lengthy sentences are often (subconsciously) assumed to be of low quality by some annotators. An extreme case is the judgement of quality through post-editing *time*: annotators have different typing speeds, as well as levels of expertise in the task of post-editing, proficiency levels in the language pair, and knowledge of the terminology used in particular sentences. These variations result in time measurements that are not comparable across annotators. Thus far, the use of post-editing time has been done on an per-annotator basis (Specia, 2011), or simply averaged across multiple translators (Plitt and Masselot, 2010), both strategies far from ideal.

Overall, these myriad of factors affecting quality judgements make the modelling of multiple annotators a very challenging problem. This problem is exacerbated when annotations are provided by non-professional annotators, e.g., through crowdsourcing – a common strategy used

to make annotation cheaper and faster, however at the cost of less reliable outcomes.

Most related work on quality assurance for data annotation has been developed in the context of crowdsourcing. Common practices include filtering out annotators who substantially deviate from a gold-standard set or present unexpected behaviours (Raykar et al., 2010; Raykar and Yu, 2012), or who disagree with others using, e.g., majority or consensus labelling (Snow et al., 2008; Sheng et al., 2008). Another relevant strand of work aims to model legitimate, systematic biases in annotators (including both non-experts and experts), such as the fact that some annotators tend to be more negative than others, and that some annotators use a wider or narrower range of values (Flach et al., 2010; Ipeirotis et al., 2010). However, with a few exceptions in Computer Vision (e.g., Whitehill et al. (2009), Welinder et al. (2010)), existing work disregard metadata and its impact on labelling.

In this paper we model the task of predicting the quality of sentence translations using datasets that have been annotated by several judges with different levels of expertise and reliability, containing translations from a variety of MT systems and on a range of different types of sentences. We address this problem using **multi-task learning** in which we learn individual models for each context (the *task*, incorporating the annotator and other metadata: translation system and the source sentence) while also modelling correlations between tasks such that related tasks can mutually inform one another. Our use of multi-task learning allows the modelling of a diversity of *truths*, while also recognising that they are rarely independent of one another (annotators often agree) by explicitly accounting for inter-annotator correlations.

Our approach is based on Gaussian Processes (GPs) (Rasmussen and Williams, 2006), a kernelised Bayesian non-parametric learning framework. We develop multi-task learning models by representing intra-task transfer simply and explicitly as part of a parameterised kernel function. GPs are an extremely flexible probabilistic framework and have been successfully adapted for multi-task learning in a number of ways, e.g., by learning multi-task correlations (Bonilla et al., 2008), modelling per-task variance (Groot et al., 2011) or per-annotator biases (Rogers et al., 2010). Our method builds on the work of Bonilla et al. (2008) by

explicitly modelling intra-task transfer, which is learned automatically from the data, in order to robustly handle outlier tasks and task variances. We show in our experiments on two translation quality datasets that these multi-task learning strategies are far superior to training individual per-task models or a single pooled model, and moreover that our multi-task learning approach can achieve similar performance to these baselines using only a fraction of the training data.

In addition to showing empirical performance gains on quality estimation applications, an important contribution of this paper is in introducing Gaussian Processes to the NLP community,<sup>1</sup> a technique that has great potential to further performance in a wider range of NLP applications. Moreover, the algorithms proposed herein can be adapted to improve future annotation efforts, and subsequent use of noisy crowd-sourced data.

## 2 Quality Estimation

Quality estimation (QE) for MT aims at providing an estimate on the quality of each translated segment – typically a sentence – without access to reference translations. Work in this area has become increasingly popular in recent years as a consequence of the widespread use of MT among real-world users such as professional translators. Examples of applications of QE include improving post-editing efficiency by filtering out low quality segments which would require more effort and time to correct than translating from scratch (Specia et al., 2009), selecting high quality segments to be published as they are, without post-editing (Soricut and Echiabi, 2010), selecting a translation from either an MT system or a translation memory for post-editing (He et al., 2010), selecting the best translation from multiple MT systems (Specia et al., 2010), and highlighting sub-segments that need revision (Bach et al., 2011).

QE is generally addressed as a machine learning task using a variety of linear and kernel-based regression or classification algorithms to induce models from examples of translations described through a number of features and annotated for quality. For an overview of various algorithms and features we refer the reader to the WMT12 shared task on QE (Callison-Burch et al., 2012).

While initial work used annotations derived

---

<sup>1</sup>We are not strictly the first, Polajnar et al. (2011) used GPs for text classification.

from automatic MT evaluation metrics (Blatz et al., 2004) such as BLEU (Papineni et al., 2002) at training time, it soon became clear that human labels result in significantly better models (Quirk, 2004). Current work at sentence level is thus based on some form of human supervision.

As typical of subjective annotation tasks, QE datasets should contain multiple annotators to lead to models that are representative. Therefore, work in QE faces all common issues regarding variability in annotators’ judgements. The following are a few other features that make our datasets particularly interesting:

- In order to minimise annotation costs, translation instances are often spread among annotators, such that each instance is only labelled by one or a few judges. In fact, for a sizeable dataset (thousands of instances), the annotation of a complete dataset by a single judge may become infeasible.
- It is often desirable to include alternative translations of source sentences produced by multiple MT systems, which requires multiple annotators for unbiased judgements, particularly for labels such as post-editing time (a translation seen a second time will require less editing effort).
- For crowd-sourced annotations it is often impossible to ensure that the same annotators will label the same subset of cases.

These features – which are also typical of many other linguistic annotation tasks – make the learning process extremely challenging. Learning models from datasets annotated by multiple annotators remains an open challenge in QE, as we show in Section 4. In what follows, we present our QE datasets in more detail.

## 2.1 Datasets

We use two freely available QE datasets to experiment with the techniques proposed in this paper:<sup>2</sup>

**WMT12:** This dataset was distributed as part of the WMT12 shared task on QE (Callison-Burch et al., 2012). It contains 1,832 instances for training, and 422 for test. The English source sentences are a subset of WMT09-12 test sets. The Spanish MT outputs were created using a standard PBSMT Moses engine. Each instance was annotated with post-editing effort scores from highest

effort (score 1) to lowest effort (score 5), where each score identifies an estimated percentage of the MT output that needs to be corrected. The post-editing effort scores were produced independently by three professional translators based on a previously post-edited translation by a fourth translator. In an attempt to accommodate for systematic biases among annotators, the final effort score was computed as the weighted average between the three PE-effort scores, with more weight given to the judges with higher standard deviation from their own mean score. This resulted in scores spread more evenly in the [1, 5] range.

**WPTP12:** This dataset was distributed by Koponen et al. (2012). It contains 299 English sentences translated into Spanish using two or more of eight MT systems randomly selected from all system submissions for WMT11 (Callison-Burch et al., 2011). These MT systems range from online and customised SMT systems to commercial rule-based systems. Translations were post-edited by humans while time was recorded. The labels are the number of seconds spent by a translator editing a sentence normalised by source sentence length. The post-editing was done by eight native speakers of Spanish, including five professional translators and three translation students. Only 20 translations were edited by all eight annotators, with the remaining translations randomly distributed amongst them. The resulting dataset contains 1,624 instances, which were randomly split into 1,300 for training and 300 for test. According to the analysis in (Koponen et al., 2012), while on average certain translators were found to be faster than others, their speed in post-editing individual sentences varies considerably, i.e., certain translators are faster at certain sentences. To our knowledge, no previous work has managed to successfully model the prediction of post-editing time from datasets with multiple annotators.

## 3 Gaussian Process Regression

Machine learning models for quality estimation typically treat the problem as regression, seeking to model the relationship between features of the text input and the human quality judgement as a continuous response variable. Popular choices include Support Vector Machines (SVMs), which have been shown to perform well for quality estimation (Callison-Burch et al., 2012) using non-linear kernel functions such as radial basis func-

<sup>2</sup>Both datasets can be downloaded from <http://www.dcs.shef.ac.uk/~lucia/resources.html>.

tions. In this paper we consider Gaussian Processes (GP) (Rasmussen and Williams, 2006), a probabilistic machine learning framework incorporating kernels and Bayesian non-parametrics, widely considered state-of-the-art for regression. Despite this GPs have not been used widely to date in statistical NLP. GPs are particularly suitable for modelling QE for a number of reasons: 1) they explicitly model uncertainty, which is rife in QE datasets; 2) they allow fitting of expressive kernels to data, in order to modulate the effect of features of varying usefulness; and 3) they can naturally be extended to model correlated tasks using multi-task kernels. We now give a brief overview of GPs, following Rasmussen and Williams (2006).

In our regression task<sup>3</sup> the data consists of  $n$  pairs  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$ , where  $\mathbf{x}_i \in \mathbb{R}^F$  is a  $F$ -dimensional feature vector and  $y_i \in \mathbb{R}$  is the response variable. Each instance is a translation and the feature vector encodes its linguistic features; the response variable is a numerical quality judgement: post editing time or likert score. As usual, the modelling challenge is to automatically predict the value of  $y$  based on the  $\mathbf{x}$  for unseen test input.

GP regression assumes the presence of a latent function,  $f : \mathbb{R}^F \rightarrow \mathbb{R}$ , which maps from the input space of feature vectors  $\mathbf{x}$  to a scalar. Each response value is then generated from the function evaluated at the corresponding data point,  $y_i = f(\mathbf{x}_i) + \eta$ , where  $\eta \sim \mathcal{N}(0, \sigma_n^2)$  is added white-noise. Formally  $f$  is drawn from a GP prior,

$$f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}')) \quad ,$$

which is parameterised by a mean (here, 0) and a covariance kernel function  $k(\mathbf{x}, \mathbf{x}')$ . The kernel function represents the covariance (i.e., similarities in the response) between pairs of data points. Intuitively, points that are in close proximity should have high covariance compared to those that are further apart, which constrains  $f$  to be a smoothly varying function of its inputs. This intuition is embodied in the squared exponential kernel (*a.k.a.* radial basis function or Gaussian),

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T A^{-1}(\mathbf{x} - \mathbf{x}')\right) \quad (1)$$

where  $\sigma_f^2$  is a scaling factor describing the overall levels of variance, and  $A = \text{diag}(\mathbf{a})$  is a diagonal

<sup>3</sup>Our approach generalises to classification, ranking (ordinal regression) or various other training objectives, including mixtures of objectives. In this paper we use regression for simplicity of exposition and implementation.

matrix of length scales, encoding the smoothness of functions  $f$  with respect to each feature. Non-uniform length scales allow for different degrees of smoothness of  $f$  in each dimension, such that e.g., for unimportant features  $f$  is relatively flat whereas for very important features  $f$  is jagged, such that a small change in the feature value has a large effect. When the values of  $\mathbf{a}$  are learned automatically from data, as we do herein, this is referred to as the *automatic relevance determination* (ARD) kernel.

Given the generative process defined above, we formulate prediction as Bayesian inference under the posterior, namely

$$p(y_* | \mathbf{x}_*, \mathcal{D}) = \int_f p(y_* | \mathbf{x}_*, f) p(f | \mathcal{D})$$

where  $\mathbf{x}_*$  is a test input and  $y_*$  is its response value. The posterior  $p(f | \mathcal{D})$  reflects our updated belief over possible functions after observing the training set  $\mathcal{D}$ , i.e.,  $f$  should pass close to the response values for each training instance (but need not fit exactly due to additive noise). This is balanced against the smoothness constraints that arise from the GP prior. The predictive posterior can be solved analytically, resulting in

$$y_* \sim \mathcal{N}(\mathbf{k}_*^T (K + \sigma_n^2 I)^{-1} \mathbf{y}, \quad (2)$$

$$k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T (K + \sigma_n^2 I)^{-1} \mathbf{k}_*)$$

where  $\mathbf{k}_* = [k(\mathbf{x}_*, \mathbf{x}_1) \ k(\mathbf{x}_*, \mathbf{x}_2) \ \dots \ k(\mathbf{x}_*, \mathbf{x}_n)]^T$  are the kernel evaluations between the test point and the training set, and  $\{K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)\}$  is the kernel (gram) matrix over the training points. Note that the posterior in Eq. 2 includes not only the expected response (the mean) but also the variance, encoding the model’s uncertainty, which is important for integration into subsequent processing, e.g., as part of a larger probabilistic model.

GP regression also permits an analytic formulation of the marginal likelihood,  $p(\mathbf{y} | X) = \int_f p(\mathbf{y} | X, f) p(f)$ , which can be used for model training ( $X$  are the training inputs). Specifically, we can derive the gradient of the (log) marginal likelihood with respect to the model hyperparameters (i.e.,  $\mathbf{a}, \sigma_n, \sigma_s$  etc.) and thereby find the type II maximum likelihood estimate using gradient ascent. Note that in general the marginal likelihood is non-convex in the hyperparameter values, and consequently the solutions may only be locally optimal. Here we bootstrap the learning of complex models with many hyperparameters by initialising

with the (good) solutions found for simpler models, thereby avoiding poor local optima. We refer the reader to Rasmussen and Williams (2006) for further details.

At first glance GPs resemble SVMs, which also admit kernels such as the popular squared exponential kernel in Eq. 1. The key differences are that GPs are probabilistic models and support exact Bayesian inference in the case of regression (approximate inference is required for classification (Rasmussen and Williams, 2006)). Moreover GPs provide greater flexibility in fitting the kernel hyperparameters even for complex composite kernels. In typical usage, the kernel hyperparameters for an SVM are fit using held-out estimation, which is inefficient and often involves tying together parameters to limit the search complexity (e.g., using a single scale parameter in the squared exponential). Multiple-kernel learning (Gönen and Alpaydm, 2011) goes some way to addressing this problem within the SVM framework, however this technique is limited to reweighting linear combinations of kernels and has high computational complexity.

### 3.1 Multi-task Gaussian Process Models

Until now we have considered a standard regression scenario, where each training point is labelled with a single output variable. In order to model multiple different annotators jointly, i.e., multi-task learning, we need to extend the model to handle many tasks. Conceptually, we can consider the multi-task model drawing a latent function for each task,  $f_m(\mathbf{x})$ , where  $m \in 1, \dots, M$  is the task identifier. This function is then used to explain the response values for all the instances for that task (subject to noise). Importantly, for multi-task learning to be of benefit, the prior over  $\{f_m\}$  must correlate the functions over different tasks, e.g., by imposing similarity constraints between the values for  $f_m(\mathbf{x})$  and  $f_{m'}(\mathbf{x})$ .

We can consider two alternative perspectives for framing the multi-task learning problem: either *isotopic* where we associate each input point  $\mathbf{x}$  with a vector of outputs,  $\mathbf{y} \in \mathbb{R}^M$ , one for each of the  $M$  tasks; or *heterotopic* where some of the outputs are missing, i.e., tasks are not constrained to share the same data points (Alvarez et al., 2011). Given the nature of our datasets, we opted for the heterotopic approach, which can handle both singly annotated and multiply annotated

data. This can be implemented by augmenting each input point with an additional task identity feature, which is paired with a single  $y$  response, and integrated into a GP model with the standard training and inference algorithms.<sup>4</sup>

In moving to a task-augmented data representation, we need to revise our kernel function. We use a separable multi-task kernel (Bonilla et al., 2008; Alvarez et al., 2011) of the form

$$k((\mathbf{x}, d), (\mathbf{x}', d')) = k^{\text{data}}(\mathbf{x}, \mathbf{x}')B_{d,d'}, \quad (3)$$

where  $k^{\text{data}}(\mathbf{x}, \mathbf{x}')$  is a standard kernel over the input points, typically a squared exponential (see Eq. 1), and  $B \in \mathbb{R}^{D \times D}$  is a positive semi-definite matrix encoding task covariances. We develop a series of increasingly complex choices for  $B$ , which we compare empirically in Section 4.2:

**Independent** The simplest case is where  $B = I$ , i.e., all pairs of different tasks have zero covariance. This corresponds to independent modelling of each task, although all models share the same data kernel, so this setting is not strictly equivalent to independent training with independent per-task data kernels (with different hyperparameters). Similarly, we might choose to use a single noise variance,  $\sigma_n^2$ , or an independent noise variance hyperparameter per task.

**Pooled** Another extreme is  $B = \mathbf{1}$ , which ignores the task identity, corresponding to pooling the multi-task data into one large set. Groot et al. (2011) present a method for applying GPs for modelling multi-annotator data using this **pooling** kernel with independent per-task noise terms. They show on synthetic data experiments that this approach works well at extracting the signal from noise-corrupted inputs.

**Combined** A simple approach for  $B$  is a weighted combination of **Independent** and **Pool**, i.e.,  $B = \mathbf{1} + aI$ , where the hyperparameter  $a \geq 0$  controls the amount of inter-task transfer between each task and the global ‘pooled’ task.<sup>5</sup> For dissimilar tasks, a high value of  $a$  allows each task to be modelled independently, while for more similar tasks low  $a$  allows the use of a large pool of

<sup>4</sup>Note that the separable kernel (Eq. 3) gives rise to block structured kernel matrices which permit various optimisations (Bonilla et al., 2008) to reduce the computational complexity of inference, e.g., the matrix inversion in Eq. 2.

<sup>5</sup>Note that larger values of  $a$  need not affect the overall magnitude of  $k$ , which can be down-scaled by the  $\sigma_f^2$  factor in the data kernel (Eq. 1).

similar data. A scaled version of this kernel has been shown to correspond to mean regularisation in SVMs when combined with a linear data kernel (Evgeniou et al., 2006). A similar multi-task kernel was proposed by Daumé III (2007), using a linear data kernel and  $a = 1$ , which has shown to result in excellent performance across a range of NLP problems. In contrast to these earlier approaches, we learn the hyperparameter  $a$  directly, fitting the relative amounts of inter- versus intra-task transfer to the dataset.

**Combined+** We consider an extension to the **Combined** kernel,  $B = \mathbf{1} + \text{diag}(\mathbf{a})$ ,  $a_d \geq 0$  in which each task has a different hyperparameter modulating its independence from the global pool. This additional flexibility can be used, e.g., to allow individual outlier annotators to be modelled independently of the others, by assigning a high value to  $a_d$ . In contrast, **Combined** ties together the parameters for all tasks, i.e., all annotators are assumed to have similar quality in that they deviate from the mean to the same degree.

### 3.2 Integrating metadata

The approaches above assume that the data is split into an unstructured set of  $M$  tasks, e.g., by annotator. However, it is often the case that we have additional information about each data instance in the form of metadata. In our quality estimation experiments we consider as metadata the MT system which produced the translation, and the identity of the source sentence being translated. Many other types of metadata, such as the level of experience of the annotator, could also be used. One way of integrating such metadata would be to define a separate task for every observed combination of metadata values, in which case we treat the metadata as a *task descriptor*. Doing so naively would however incur a significant penalty, as each task will have very few training instances resulting in inaccurate models, even with the inter-task kernel approaches defined above.

We instead extend the task-level kernels to use the task descriptors directly to represent task correlations. Let  $B^{(i)}$  be a square covariance matrix for the  $i^{\text{th}}$  task descriptor of  $M$ , with a column and row for each value (e.g., annotator identity, translation system, etc.). We redefine the task level kernel using paired inputs  $(\mathbf{x}, \mathbf{m})$ , where  $\mathbf{m}$  are the

task descriptors,

$$k((\mathbf{x}, \mathbf{m}), (\mathbf{x}', \mathbf{m}')) = k^{\text{data}}(\mathbf{x}, \mathbf{x}') \prod_{i=1}^M B_{m_i, m'_i}^{(i)}.$$

This is equivalent to using a structured task-kernel  $B = B^{(1)} \otimes B^{(2)} \otimes \dots \otimes B^{(M)}$  where  $\otimes$  is the Kronecker product. Using this formulation we can consider any of the above choices for  $B$  applied to each task descriptor. In our experiments we consider the **Combined** and **Combined+** kernels, which allow the model to learn the relative importance of each descriptor in terms of independent modelling versus pooling the data.

## 4 Multi-task Quality Estimation

### 4.1 Experimental Setup

**Feature sets:** In all experiments we use 17 shallow QE features that have been shown to perform well in previous work. These were used by a highly competitive baseline entry in the WMT12 shared task, and were extracted here using the system provided by that shared task.<sup>6</sup> They include simple counts, e.g., the tokens in sentences, as well as source and target language model probabilities. Each feature was scaled to have zero mean and unit standard deviation on the training set.

**Baselines:** The baselines use the SVM regression algorithm with radial basis function kernel and parameters  $\gamma$ ,  $\epsilon$  and  $C$  optimised through grid-search and 5-fold cross validation on the training set. This is generally a very strong baseline: in the WMT12 QE shared task, only five out of 19 submissions were able to significantly outperform it, and only by including many complex additional features, tree kernels, etc. We also present  $\mu$ , a trivial baseline based on predicting for each test instance the training mean (overall, and for specific tasks).

**GP:** All GP models were implemented using the GPML Matlab toolbox.<sup>7</sup> Hyperparameter optimisation was performed using conjugate gradient ascent of the log marginal likelihood function, with up to 100 iterations. The simpler models were initialised with all hyperparameters set to one, while more complex models were initialised using the

<sup>6</sup>The software used to extract these (and other) features can be downloaded from <http://www.quest.dcs.shef.ac.uk/>

<sup>7</sup><http://www.gaussianprocess.org/gpml/code>

Model	MAE	RMSE
$\mu$	0.8279	0.9899
SVM	0.6889	0.8201
Linear ARD	0.7063	0.8480
Squared exp. Isotropic	0.6813	0.8146
Squared exp. ARD	<b>0.6680</b>	<b>0.8098</b>
Rational quadratic ARD	0.6773	0.8238
Matern(5,2)	0.6772	0.8124
Neural network	0.6727	0.8103

Table 1: Single-task learning results on the WMT12 dataset, trained and evaluated against the weighted averaged response variable.  $\mu$  is a baseline which predicts the training mean, SVM uses the same system as the WMT12 QE task, and the remainder are GP regression models with different kernels (all include additive noise).

solution for a simpler model. For instance, models using ARD kernels were initialised from an equivalent isotropic kernel (which ties all the hyperparameters together), and independent per-task noise models were initialised from a single noise model. This approach was more reliable than random restarts in terms of accuracy and runtime efficiency.

**Evaluation:** We evaluate predictive accuracy using two measures: mean absolute error,  $MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$  and root mean square error,  $RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$ , where  $y_i$  are the gold standard response values and  $\hat{y}_i$  are the model predictions.

## 4.2 Results

Our experiments aim to demonstrate the efficacy of GP regression, both the single task and multi-task settings, compared to competitive baselines.

**WMT12: Single task** We start by comparing GP regression with alternative approaches using the WMT12 dataset on the standard task of predicting a weighted mean quality rating (as it was done in the WMT12 QE shared task). Table 1 shows the results for baseline approaches and the GP models, using a variety of different kernels (see Rasmussen and Williams (2006) for details of the kernel functions). From this we can see that all models do much better than the mean baseline and that most of the GP models have lower error than the state-of-the-art SVM. In terms of kernels, the linear kernel performs comparatively worse than non-linear kernels. Overall the squared exponen-

Model	MAE	RMSE
$\mu$	0.8541	1.0119
Independent SVMs	0.7967	0.9673
EasyAdapt SVM	0.7655	0.9105
Independent	0.7061	0.8534
Pooled	0.7252	0.8754
Pooled & {N}	0.7050	0.8497
Combined	<b>0.6966</b>	<b>0.8448</b>
Combined & {N}	0.6975	0.8476
Combined+	0.6975	0.8463
Combined+ & {N}	0.7046	0.8595

Table 2: Results on the WMT12 dataset, trained and evaluated over all three annotator’s judgements. Shown above are the training mean baseline  $\mu$ , single-task learning approaches, and multi-task learning models, with the columns showing macro average error rates over all three response values. All systems use a squared exponential ARD kernel in a product with the named task-kernel, and with added noise (per-task noise is denoted {N}, otherwise has shared noise).

tial ARD kernel has the best performance under both measures of error, and for this reason we use this kernel in our subsequent experiments.

**WMT12: Multi-task** We now turn to the multi-task setting, where we seek to model each of the three annotators’ predictions. Table 2 presents the results. Note that here error rates are measured over all of the three annotators’ judgements, and consequently are higher than those measured against their average response in Table 1. For comparison, taking the predictions of the best model, **Combined**, in Table 2 and evaluating its averaged prediction has a MAE of 0.6588 vs. the averaged gold standard, significantly outperforming the best model in Table 1.

There are a number of important findings in Table 2. First, the independently trained models do well, outperforming the pooled model with fixed noise, indicating that naively pooling the data is counter-productive and that there are annotator-specific biases. Including per-annotator noise to the pooled model provides a boost in performance, however the best results are obtained using the **Combined** kernel which brings the strengths of both the independent and pooled settings. There are only minor differences between the different multi-task kernels, and in this case per-annotator noise made little difference. An explanation for the contradictory findings about the importance

of independent noise is that differences between annotators can already be explained by the MTL model using the multi-task kernel, and need not be explained as noise.

The GP models significantly improve over the baselines, including an SVM trained independently and using the EasyAdapt method for multi-task learning (Daumé III, 2007). While EasyAdapt showed an improvement over the independent SVM, it was a long way short of the GP models. A possible explanation is that in EasyAdapt the multi-task sharing parameter is set at  $a = 1$ , which may not be appropriate for the task. In contrast the **Combined** GP model learned a value of  $a = 0.01$ , weighting the value of pooling much more highly than independent training.

A remaining question is how these approaches cope with smaller datasets, where issues of data sparsity become more prevalent. To test this, we trained single-task, pooled and multi-task models on randomly sub-sampled training sets of different sizes, and plot their error rates in Figure 1. As expected, for very small datasets pooling outperforms single task learning, however for modest sized datasets of  $\geq 90$  training instances pooling was inferior. For all dataset sizes multi-task learning is superior to the other approaches, making much better use of small and large training sets. The MTL model trained on 500 samples had an MAE of  $0.7082 \pm 0.0042$ , close to the best results from the full dataset in Table 2, despite using  $\frac{1}{9}$  as much data: here we use  $\frac{1}{3}$  as many training instances where each is singly (cf. triply) annotated. The same experiments run with multiply-annotated instances showed much weaker performance, presumably due to the more limited sample of input points and poorer fit of the ARD kernel hyperparameters. This finding suggests that our multi-task learning approach could be used to streamline annotation efforts by reducing the need for extensive multiple annotations.

**WPTP12** This dataset involves predicting the post-editing time for eight annotators, where we seek to test our model’s capability to use additional metadata. We model the logarithm of the per-word post-editing time, in order to make the response variable more comparable between annotators and across sentences, and generally more Gaussian in shape. In Table 3 immediately we can see that the baseline of predicting the training mean is very difficult to beat, and the trained

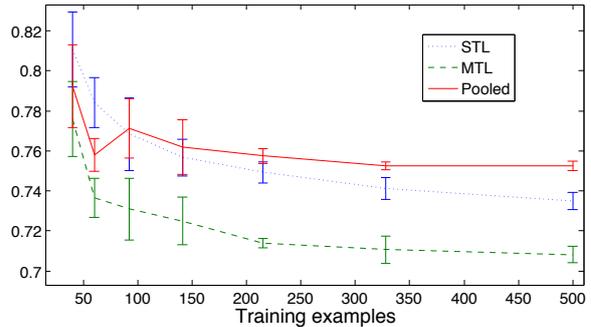


Figure 1: Learning curve comparing MAE for different training methods on the **WMT12** dataset, all using a squared exponential ARD data kernel and tied noise parameter. The MTL model uses the **Combined** task kernel. Each point is the average of 5 runs, and the error bars show  $\pm 1$  s.d.

systems often do worse. Partitioning the data by annotator ( $\mu_A$ ) gives the best baseline result, while there is less information from the MT system or sentence identity. Single-task learning performs only a little better than these baselines, although some approaches such as the naive pooling perform terribly. This suggests that the tasks are highly different to one another. Interestingly, adding the per-task noise models to the pooling approach greatly improves its performance.

The multi-task learning methods performed best when using the annotator identity as the task descriptor, and less well for the MT system and sentence pair, where they only slightly improved over the baseline. However, making use of all these layers of metadata together gives substantial further improvements, reaching the best result with **Combined**<sub>A,S,T</sub>. The effect of adding per-task noise to these models was less marked than for the pooled models, as in the WMT12 experiments. Inspecting the learned hyperparameters, the combined models learned a large bias towards independent learning over pooling, in contrast to the WMT12 experiments. This may explain the poor performance of EasyAdapt on this dataset.

## 5 Conclusion

This paper presented a novel approach for learning from human linguistic annotations by explicitly training models of individual annotators (and possibly additional metadata) using multi-task learning. Our method using Gaussian Processes is flexible, allowing easy learning of inter-dependences between different annotators and other task meta-

Model	MAE	RMSE
$\mu$	0.5596	0.7053
$\mu_A$	0.5184	0.6367
$\mu_S$	0.5888	0.7588
$\mu_T$	0.6300	0.8270
Pooled SVM	0.5823	0.7472
Independent <sub>A</sub> SVM	0.5058	0.6351
EasyAdapt SVM	0.7027	0.8816
SINGLE-TASK LEARNING		
Independent <sub>A</sub>	0.5091	0.6362
Independent <sub>S</sub>	0.5980	0.7729
Pooled	0.5834	0.7494
Pooled & {N}	0.4932	0.6275
MULTI-TASK LEARNING: <i>Annotator</i>		
Combined <sub>A</sub>	0.4815	0.6174
Combined <sub>A</sub> & {N}	0.4909	0.6268
Combined+ <sub>A</sub>	0.4855	0.6203
Combined+ <sub>A</sub> & {N}	0.4833	0.6102
MULTI-TASK LEARNING: <i>Translation system</i>		
Combined <sub>S</sub>	0.5825	0.7482
MULTI-TASK LEARNING: <i>Sentence pair</i>		
Combined <sub>T</sub>	0.5813	0.7410
MULTI-TASK LEARNING: <i>Combinations</i>		
Combined <sub>A,S</sub>	0.4988	0.6490
Combined <sub>A,S</sub> & {N <sub>A,S</sub> }	0.4707	0.6003
Combined+ <sub>A,S</sub>	0.4772	0.6094
Combined <sub>A,S,T</sub>	<b>0.4588</b>	<b>0.5852</b>
Combined <sub>A,S,T</sub> & {N <sub>A,S</sub> }	0.4723	0.6023

Table 3: Results on the **WPTP12** dataset, using the log of the post-editing time per word as the response variable. Shown above are the training mean and SVM baselines, single-task learning and multi-task learning results (micro average). The subscripts denote the task split: annotator (A), MT system (S) and sentence identity (T).

data. Our experiments showed how our approach outperformed competitive baselines on two machine translation quality regression problems, including the highly challenging problem of predicting post-editing time.

In future work we plan to apply these techniques to new datasets, particularly noisy crowd-sourced data with much large numbers of annotators, as well as a wider range of task types and mixtures thereof (regression, ordinal regression, ranking, classification). We also have preliminary positive results for more advanced multi-task kernels, e.g., general dense matrices, which can induce clusters of related tasks.

Our multi-task learning approach has much wider application. Models of individual annotators could be used to train machine translation systems to optimise an annotator-specific quality measure, or in active learning for corpus annotation, where the model can suggest the most appropriate instances for each annotator or the best annotator for a given instance. Further, our approach contributes to work based on cheap and fast crowdsourcing of linguistic annotation by minimising the need for careful data curation and quality control.

## Acknowledgements

This work was funded by PASCAL2 Harvest Programme, as part of the QuEst project: <http://www.quest.dcs.shef.ac.uk/>. The authors would like to thank Neil Lawrence and James Hensman for advice on Gaussian Processes, the QuEst participants, particularly José Guilherme Camargo de Souza and Eva Hassler, and the three anonymous reviewers.

## References

- Mauricio A. Alvarez, Lorenzo Rosasco, and Neil D. Lawrence. 2011. Kernels for vector-valued functions: A review. *Foundations and Trends in Machine Learning*, 4(3):195–266.
- Nguyen Bach, Fei Huang, and Yaser Al-Onaizan. 2011. Goodness: a method for measuring machine translation confidence. In *the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 211–219, Portland, Oregon.
- John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto

- Sanchis, and Nicola Ueffing. 2004. Confidence Estimation for Machine Translation. In *the 20th International Conference on Computational Linguistics (Coling 2004)*, pages 315–321, Geneva.
- Edwin Bonilla, Kian Ming Chai, and Christopher Williams. 2008. Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems (NIPS)*.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada.
- Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic.
- Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. 2006. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6(1):615.
- Peter A. Flach, Sebastian Spiegler, Bruno Golénia, Simon Price, John Guiver, Ralf Herbrich, Thore Graepel, and Mohammed J. Zaki. 2010. Novel tools to streamline the conference review process: experiences from SIGKDD’09. *SIGKDD Explor. Newsl.*, 11(2):63–67, May.
- Mehmet Gönen and Ethem Alpaydm. 2011. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268.
- Perry Groot, Adriana Birlutiu, and Tom Heskes. 2011. Learning from multiple annotators with gaussian processes. In *Proceedings of the 21st international conference on Artificial neural networks - Volume Part II, ICANN’11*, pages 159–164, Espoo, Finland.
- Yifan He, Yanjun Ma, Josef van Genabith, and Andy Way. 2010. Bridging smt and tm with translation recommendation. In *the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden.
- Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation, HCOMP ’10*, pages 64–67, Washington DC.
- Maarit Koponen, Wilker Aziz, Luciana Ramos, and Lucia Specia. 2012. Post-editing time as a measure of cognitive effort. In *Proceedings of the AMTA 2012 Workshop on Post-editing Technology and Practice, WPTP 2012*, San Diego, CA.
- Kishore Papineni, Salim Roukos, Todd Ward, and Weijing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania.
- Mirko Plitt and François Masselot. 2010. A productivity test of statistical machine translation post-editing in a typical localisation context. *Prague Bull. Math. Linguistics*, 93:7–16.
- Tamara Polajnar, Simon Rogers, and Mark Girolami. 2011. Protein interaction detection in sentences via gaussian processes; a preliminary evaluation. *Int. J. Data Min. Bioinformatics*, 5(1):52–72, February.
- Christopher B. Quirk. 2004. Training a sentence-level machine translation confidence metric. In *Proceedings of the International Conference on Language Resources and Evaluation*, volume 4 of *LREC 2004*, pages 825–828, Lisbon, Portugal.
- Carl E. Rasmussen and Christopher K.I. Williams. 2006. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA.
- Vikas C. Raykar and Shipeng Yu. 2012. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *J. Mach. Learn. Res.*, 13:491–518.
- Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *J. Mach. Learn. Res.*, 99:1297–1322.
- Simon Rogers, Mark Girolami, and Tamara Polajnar. 2010. Semi-parametric analysis of multi-rater data. *Statistics and Computing*, 20(3):317–334.
- Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD, KDD’08*, pages 614–622, Las Vegas, Nevada.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In *the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii.
- Radu Soricut and Abdessamad Echihabi. 2010. Trustrank: Inducing trust in automatic translations via ranking. In *the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 612–621, Uppsala, Sweden, July.
- Lucia Specia, Marco Turchi, Nicola Cancedda, Marc Dymetman, and Nello Cristianini. 2009. Estimating the Sentence-Level Quality of Machine Translation Systems. In *the 13th Annual Meeting of the European Association for Machine Translation (EAMT’2009)*, pages 28–37, Barcelona.

- Lucia Specia, Dhwanj Raj, and Marco Turchi. 2010. Machine translation evaluation versus quality estimation. *Machine Translation*, pages 39–50.
- Lucia Specia. 2011. Exploiting Objective Annotations for Measuring Translation Post-editing Effort. In *the 15th Annual Meeting of the European Association for Machine Translation (EAMT'2011)*, pages 73–80, Leuven.
- Peter Welinder, Steve Branson, Serge Belongie, and Pietro Perona. 2010. The Multidimensional Wisdom of Crowds. In *Advances in Neural Information Processing Systems*, volume 23, pages 2424–2432.
- Jacob Whitehill, Paul Ruvolo, Ting-fan Wu, Jacob Bergsma, and Javier Movellan. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22:2035–2043.

# Smoothed marginal distribution constraints for language modeling

Brian Roark<sup>†</sup>, Cyril Allauzen<sup>°</sup> and Michael Riley<sup>°</sup>

<sup>†</sup>Oregon Health & Science University, Portland, Oregon    <sup>°</sup>Google, Inc., New York  
roarkbr@gmail.com, {allauzen,riley}@google.com

## Abstract

We present an algorithm for re-estimating parameters of backoff n-gram language models so as to preserve given marginal distributions, along the lines of well-known Kneser-Ney (1995) smoothing. Unlike Kneser-Ney, our approach is designed to be applied to any given smoothed backoff model, including models that have already been heavily pruned. As a result, the algorithm avoids issues observed when pruning Kneser-Ney models (Siivola et al., 2007; Chelba et al., 2010), while retaining the benefits of such marginal distribution constraints. We present experimental results for heavily pruned backoff n-gram models, and demonstrate perplexity and word error rate reductions when used with various baseline smoothing methods. An open-source version of the algorithm has been released as part of the OpenGrm ngram library.<sup>1</sup>

## 1 Introduction

Smoothed n-gram language models are the de-facto standard statistical models of language for a wide range of natural language applications, including speech recognition and machine translation. Such models are trained on large text corpora, by counting the frequency of n-gram collocations, then normalizing and smoothing (regularizing) the resulting multinomial distributions. Standard techniques store the observed n-grams and derive probabilities of unobserved n-grams via their longest observed suffix and “backoff” costs associated with the prefix histories of the unobserved suffixes. Hence the size of the model grows with the number of observed n-grams, which is very large for typical training corpora.

Natural language applications, however, are commonly used in scenarios requiring relatively small footprint models. For example, applications running on mobile devices or in low latency streaming scenarios may be required to limit the complexity of models and algorithms to achieve the desired operating profile. As a result, statistical language models – an important component of many such applications – are often trained on very large corpora, then modified to fit within some pre-specified size bound. One method to achieve significant space reduction is through randomized data structures, such as Bloom (Talbot and Osborne, 2007) or Bloomier (Talbot and Brants, 2008) filters. These data structures permit efficient querying for specific n-grams in a model that has been stored in a fraction of the space required to store the full, exact model, though with some probability of false positives. Another common approach – which we pursue in this paper – is model pruning, whereby some number of the n-grams are removed from explicit storage in the model, so that their probability must be assigned via backoff smoothing. One simple pruning method is count thresholding, i.e., discarding n-grams that occur less than  $k$  times in the corpus. Beyond count thresholding, the most widely used pruning methods (Seymore and Rosenfeld, 1996; Stolcke, 1998) employ greedy algorithms to reduce the number of stored n-grams by comparing the stored probabilities to those that would be assigned via the backoff smoothing mechanism, and removing those with the least impact according to some criterion.

While these greedy pruning methods are highly effective for models estimated with most common smoothing approaches, they have been shown to be far less effective with Kneser-Ney trained language models (Siivola et al., 2007; Chelba et al., 2010), leading to severe degradation in model quality relative to other standard smoothing meth-

<sup>1</sup>[www.opengrm.org](http://www.opengrm.org)

4-gram models Smoothing method	Backoff			Interpolated		
	Perplexity full	pruned	n-grams ( $\times 1000$ )	Perplexity full	pruned	n-grams ( $\times 1000$ )
Absolute Discounting (Ney et al., 1994)	120.5	197.3	383.4	119.8	198.1	386.2
Witten-Bell (Witten and Bell, 1991)	118.8	196.3	380.4	121.6	202.3	396.4
Ristad (1995)	126.4	203.6	395.6	----- N/A -----		
Katz (1987)	119.8	198.1	386.2	----- N/A -----		
Kneser-Ney (Kneser and Ney, 1995)	114.5	285.1	388.2	115.8	274.3	398.7
Mod. Kneser-Ney (Chen and Goodman, 1998)	116.3	280.6	396.2	112.8	270.7	399.1

Table 1: Reformatted version of Table 3 in Chelba et al. (2010), demonstrating perplexity degradation of Kneser-Ney smoothed models in contrast to other common smoothing methods. Data: English Broadcast News, 128M words training; 692K words test; 143K word vocabulary. 4-gram language models, pruned using Stolcke (1998) relative entropy pruning to approximately 1.3% of the original size of 31,095,260 n-grams.

ods. Thus, while Kneser-Ney may be the preferred smoothing method for large, unpruned models – where it can achieve real improvements over other smoothing methods – when relatively sparse, pruned models are required, it has severely diminished utility.

Table 1 presents a slightly reformatted version of Table 3 from Chelba et al. (2010). In their experiments (see Table 1 caption for specifics on training/test setup), they trained 4-gram Broadcast News language models using a variety of both backoff and interpolated smoothing methods and measured perplexity before and after Stolcke (1998) relative entropy based pruning. With this size training data, the perplexity of all of the smoothing methods other than Kneser-Ney degrades from around 120 with the full model to around 200 with the heavily pruned model. Kneser-Ney smoothed models have lower perplexity with the full model than the other methods by about 5 points, but degrade with pruning to far higher perplexity, between 270-285.

The cause of this degradation is Kneser-Ney’s unique method for estimating smoothed language models, which will be presented in more detail in Section 3. Briefly, the smoothing method reestimates lower-order n-gram parameters in order to avoid over-estimating the likelihood of n-grams that already have ample probability mass allocated as part of higher-order n-grams. This is done via a marginal distribution constraint which requires the expected frequency of the lower-order n-grams to match their observed frequency in the training data, much as is commonly done for maximum entropy model training. Goodman (2001) proved that, under certain assumptions, such constraints can only improve language models. Lower-order n-gram parameters resulting from Kneser-Ney are not relative frequency estimates, as with other smoothing methods; rather they are parameters

estimated specifically for use within the larger smoothed model.

There are (at least) a couple of reasons why such parameters do not play well with model pruning. First, the pruning methods commonly use lower order n-gram probabilities to derive an estimate of state marginals, and, since these parameters are no longer smoothed relative frequency estimates, they do not serve that purpose well. For this reason, the widely-used SRILM toolkit recently provided switches to modify their pruning algorithm to use another model for state marginal estimates (Stolcke et al., 2011). Second, and perhaps more importantly, the marginal constraints that were applied prior to smoothing will not in general be consistent with the much smaller pruned model. For example, if a bigram parameter is modified due to the presence of some set of trigrams, and then some or all of those trigrams are pruned from the model, the bigram associated with the modified parameter will be unlikely to have an overall expected frequency equal to its observed frequency anymore. As a result, the resulting model degrades dramatically with pruning.

In this paper, we present an algorithm that imposes marginal distribution constraints of the sort used in Kneser-Ney modeling on arbitrary smoothed backoff n-gram language models. Our approach makes use of the same sort of derivation as the original Kneser-Ney modeling, but, among other differences, relies on smoothed estimates of the empirical relative frequency rather than the unsmoothed observed frequency. The algorithm can be applied after the smoothed model has been pruned, hence avoiding the pitfalls associated with Kneser-Ney modeling. Furthermore, while Kneser-Ney is conventionally defined as a variant of absolute discounting, our method can be applied to models smoothed with any backoff smoothing, including mixtures of models, widely

used for domain adaptation.

We next establish formal preliminaries and our smoothed marginal distribution constraints method.

## 2 Preliminaries

N-gram language models are typically presented mathematically in terms of words  $w$ , the strings (histories)  $h$  that precede them, and the suffixes of the histories (backoffs)  $h'$  that are used in the smoothing recursion. Let  $V$  be a vocabulary (alphabet), and  $V^*$  a string of zero or more symbols drawn from  $V$ . Let  $V^k$  denote the set of strings  $\mathbf{w} \in V^*$  of length  $k$ , i.e.,  $|\mathbf{w}| = k$ . We will use variables  $u, v, w, x, y, z \in V$  to denote single symbols from the vocabulary;  $h, g \in V^*$  to denote history sequences preceding the specific word; and  $h', g' \in V^*$  the respective backoff histories of  $h$  and  $g$  as typically defined (see below). For a string  $\mathbf{w} = w_1 \dots w_{|\mathbf{w}|}$  we can calculate the smoothed conditional probability of each word  $w_i$  in the sequence given the  $k$  words that preceded it, depending on the order of the Markov model. Let  $h_i^k = w_{i-k} \dots w_{i-1}$  be the previous  $k$  words in the sequence. Then the smoothed model is defined recursively as follows:

$$P(w_i | h_i^k) = \begin{cases} \bar{P}(w_i | h_i^k) & \text{if } c(h_i^k w_i) > 0 \\ \alpha(h_i^k) P(w_i | h_i^{k-1}) & \text{otherwise} \end{cases}$$

where  $c(h_i^k w_i)$  is the count of the n-gram sequence  $w_{i-k} \dots w_i$  in the training corpus;  $\bar{P}$  is a regularized probability estimate that provides some probability mass for unobserved n-grams; and  $\alpha(h_i^k)$  is a factor that ensures normalization. Note that for  $h = h_i^k$ , the typically defined backoff history  $h' = h_i^{k-1}$ , i.e., the longest suffix of  $h$  that is not  $h$  itself. When we use  $h'$  and  $g'$  (for notational convenience) in future equations, it is this definition that we are using.

There are many ways to estimate  $\bar{P}$ , including absolute discounting (Ney et al., 1994), Katz (1987) and Witten and Bell (1991). Interpolated models are special cases of this form, where the  $\bar{P}$  is determined using model mixing, and the  $\alpha$  parameter is exactly the mixing factor value for the lower order model.

N-gram language models allow for a sparse representation, so that only a subset of the possible n-grams must be explicitly stored. Probabilities for the rest of the n-grams are calculated through the “otherwise” semantics in the equation above. For

an n-gram language model  $G$ , we will say that an n-gram  $hw \in G$  if it is explicitly represented in the model; otherwise  $hw \notin G$ . In the standard n-gram formulation above, the assumption is that if  $c(h_i^k w_i) > 0$  then the n-gram has a parameter; yet with pruning, we remove many observed n-grams from the model, hence this is no longer the appropriate criterion. We reformulate the standard equation as follows:

$$P(w_i | h_i^k) = \begin{cases} \beta(h_i^k w_i) & \text{if } h_i^k w_i \in G \\ \alpha(h_i^k, h_i^{k-1}) P(w_i | h_i^{k-1}) & \text{otherwise} \end{cases} \quad (1)$$

where  $\beta(h_i^k w_i)$  is the parameter associated with the n-gram  $h_i^k w_i$  and  $\alpha(h_i^k, h_i^{k-1})$  is the backoff cost associated with going from state  $h_i^k$  to state  $h_i^{k-1}$ . We assume that, if  $hw \in G$  then all prefixes and suffixes of  $hw$  are also in  $G$ .

Figure 1 presents a schema of an automaton representation of an n-gram model, of the sort used in the OpenGrm library (Roark et al., 2012). States represent histories  $h$ , and the words  $w$ , whose probabilities are conditioned on  $h$ , label the arcs, leading to the history state for the subsequent word. State labels are provided in Figure 1 as a convenience, to show the (implicit) history encoded by the state, e.g., ‘xyz’ indicates that the state represents a history with the previous three symbols being x, y and z. Failure arcs, labeled with a  $\phi$  in Figure 1, encode an “otherwise” semantics and have as destination the origin state’s backoff history. Many higher order states will back off to the same lower order state, specifically those that share the same suffix.

Note that, in general, the recursive definition of backoff may require the traversal of several back-

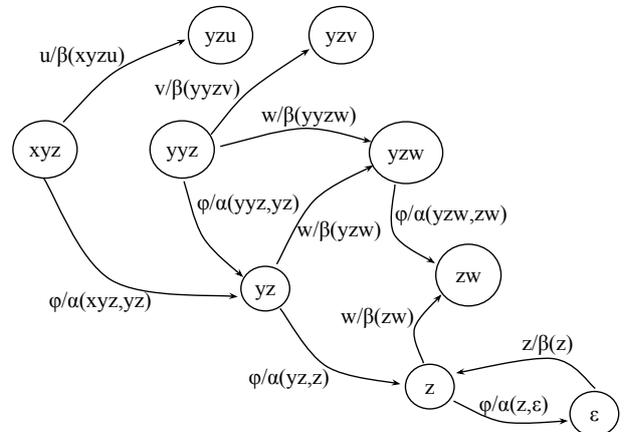


Figure 1: N-gram weighted automaton schema. State labels are presented for convenience, to specify the history implicitly encoded by the state.

off arcs before emitting a word, e.g., the highest order states in Figure 1 needing to traverse a couple of  $\phi$  arcs to reach state ‘z’. We can define the backoff cost between a state  $h_i^k$  and any of its suffix states as follows. Let  $\alpha(h, h) = 1$  and for  $m > 1$ ,

$$\alpha(h_i^k, h_i^{k-m}) = \prod_{j=1}^m \alpha(h_i^{k-j+1}, h_i^{k-j}).$$

If  $h_i^k w \notin G$  then the probability of that n-gram will be defined in terms of backoff to its longest suffix  $h_i^{k-m} w \in G$ . Let  $h^{wG}$  denote the longest suffix of  $h$  such that  $h^{wG} w \in G$ . Note that this is not necessarily a proper suffix, since  $h^{wG}$  could be  $h$  itself or it could be  $\epsilon$ . Then

$$P(w | h) = \alpha(h, h^{wG}) \beta(h^{wG} w) \quad (2)$$

which is equivalent to equation 1.

### 3 Marginal distribution constraints

Marginal distribution constraints attempt to match the expected frequency of an n-gram with its observed frequency. In other words, if we use the model to randomly generate a very large corpus, the n-grams should occur with the same relative frequency in both the generated and original (training) corpus. Standard smoothing methods overgenerate lower-order n-grams. Using standard n-gram notation (where  $g'$  is the backoff history for  $g$ ), this constraint is stated in Kneser and Ney (1995) as

$$\hat{P}(w | h') = \sum_{g:g'=h'} P(g, w | h') \quad (3)$$

where  $\hat{P}$  is the empirical relative frequency estimate. Taking this approach, certain base smoothing methods end up with very nice, easy to calculate solutions based on counts. Absolute discounting (Ney et al., 1994) in particular, using the above approach, leads to the well-known Kneser-Ney smoothing approach (Kneser and Ney, 1995; Chen and Goodman, 1998). We will follow this same approach, with a couple of changes. First, we will make use of regularized estimates of relative frequency  $\bar{P}$  rather than raw relative frequency  $\hat{P}$ . Second, rather than just looking at observed histories  $h$  that back off to  $h'$ , we will look at all histories (observed or not) of the length of the longest history in the model. For notational simplicity, suppose we have an  $n+1$ -gram model,

hence the longest history in the model is of length  $n$ . Assume the length of the particular backoff history  $|h'| = k$ . Let  $V^{n-k}h'$  be the set of strings  $h \in V^n$  with  $h'$  as a suffix. Then we can restate the marginal distribution constraint in equation 3 as

$$\bar{P}(w | h') = \sum_{h \in V^{n-k}h'} P(h, w | h') \quad (4)$$

Next we solve for  $\beta(h'w)$  parameters used in equation 1. Note that  $h'$  is a suffix of any  $h \in V^{n-k}h'$ , so conditioning probabilities on  $h$  and  $h'$  is the same as conditioning on just  $h$ . Each of the following derivation steps simply relies on the chain rule or definition of conditional probability, as well as pulling terms out of the summation.

$$\begin{aligned} \bar{P}(w | h') &= \sum_{h \in V^{n-k}h'} P(h, w | h') \\ &= \sum_{h \in V^{n-k}h'} P(w | h, h') P(h | h') \\ &= \sum_{h \in V^{n-k}h'} P(w | h) \frac{P(h)}{\sum_{g \in V^{n-k}h'} P(g)} \\ &= \frac{1}{\sum_{g \in V^{n-k}h'} P(g)} \sum_{h \in V^{n-k}h'} P(w | h) P(h) \quad (5) \end{aligned}$$

Then, multiplying both sides by the normalizing denominator on the right-hand side and using equation 2 to substitute  $\alpha(h, h^{wG}) \beta(h^{wG} w)$  for  $P(w | h)$ :

$$\begin{aligned} \bar{P}(w | h') \sum_{g \in V^{n-k}h'} P(g) &= \sum_{h \in V^{n-k}h'} P(w | h) P(h) \\ &= \sum_{h \in V^{n-k}h'} \alpha(h, h^{wG}) \beta(h^{wG} w) P(h) \quad (6) \end{aligned}$$

Note that we are only interested in  $h'w \in G$ , hence there are two disjoint subsets of histories  $h \in V^{n-k}h'$  that are being summed over: those such that  $h^{wG} = h'$  and those such that  $|h^{wG}| > |h'|$ . We next separate these sums in the next step of the derivation:

$$\begin{aligned} \bar{P}(w | h') \sum_{g \in V^{n-k}h'} P(g) &= \\ &= \sum_{h \in V^{n-k}h': |h^{wG}| > |h'|} \alpha(h, h^{wG}) \beta(h^{wG} w) P(h) + \\ &= \sum_{h \in V^{n-k}h': h^{wG} = h'} \alpha(h, h') \beta(h'w) P(h) \quad (7) \end{aligned}$$

Finally, we solve for  $\beta(h'w)$  in the second sum on the right-hand side of equation 7, yielding the formula in equation 8. Note that this equation is the correlate of equation (6) in Kneser and Ney

$$\beta(h'w) = \frac{\bar{P}(w | h') \sum_{g \in V^{n-k}h'} P(g) - \sum_{h \in V^{n-k}h': |h^{wG}| > |h'|} \alpha(h, h^{wG}) \beta(h^{wG}w) P(h)}{\sum_{h \in V^{n-k}h': h^{wG}=h'} \alpha(h, h') P(h)} \quad (8)$$

(1995), modulo the two differences noted earlier: use of smoothed probability  $\bar{P}$  rather than raw relative frequency; and summing over all history substrings in  $V^{n-k}h'$  rather than just those with count greater than zero, which is also a change due to smoothing. Keep in mind,  $\bar{P}$  is the target expected frequency from a given smoothed model. Kneser-Ney models are not useful input models, since their  $\bar{P}$  n-gram parameters are not relative frequency estimates. This means that we cannot simply ‘repair’ pruned Kneser-Ney models, but must use other smoothing methods where the smoothed values are based on relative frequency estimation.

There are, in addition, two other important differences in our approach from that in Kneser and Ney (1995), which would remain as differences even if our target expected frequency were the unsmoothed relative frequency  $\hat{P}$  instead of the smoothed estimate  $\bar{P}$ . First, the sum in the numerator is over histories of length  $n$ , the highest order in the n-gram model, whereas in the Kneser-Ney approach the sum is over histories that immediately back off to  $h'$ , i.e., from the next highest order in the n-gram model. Thus the unigram distribution is with respect to the bigram model, the bigram model is with respect to the trigram model, and so forth. In our optimization, we sum instead over all possible history sequences of length  $n$ . Second, an early assumption made in Kneser and Ney (1995) is that the denominator term in their equation (6) (our Eq. 8) is constant across all words for a given history, which is clearly false. We do not make this assumption. Of course, the probabilities must be normalized, hence the final values of  $\beta(h'w)$  will be proportional to the values in Eq. 8.

We briefly note that, like Kneser-Ney, if the baseline smoothing method is consistent, then the amount of smoothing in the limit will go to zero and our resulting model will also be consistent.

The smoothed relative frequency estimate  $\bar{P}$  and higher order  $\beta$  values on the right-hand side of Eq. 8 are given values (from the input smoothed model and previous stages in the algorithm, respectively), implying an algorithm that estimates highest orders of the model first. In addition, steady state

history probabilities  $P(h)$  must be calculated. We turn to the estimation algorithm next.

## 4 Model constraint algorithm

Our algorithm takes a smoothed backoff n-gram language model in an automaton format (see Figure 1) and returns a smoothed backoff n-gram language model with the same topology. For all n-grams in the model that are suffixes of other n-grams in the model – i.e., that are backed-off to – we calculate the weight provided by equation 8 and assign it (after normalization) to the appropriate n-gram arc in the automaton. There are several important considerations for this algorithm, which we address in this section. First, we must provide a probability for every state in the model. Second, we must memoize summed values that are used repeatedly. Finally, we must iterate the calculation of certain values that depend on the n-gram weights being re-estimated.

### 4.1 Steady state probability calculation

The steady state probability  $P(h)$  is taken to be the probability of observing  $h$  after a long word sequence, i.e., the state’s relative frequency in a long sequence of randomly-generated sentences from the model:

$$P(h) = \lim_{m \rightarrow \infty} \sum_{w_1 \dots w_m} \hat{P}(w_1 \dots w_m h) \quad (9)$$

where  $\hat{P}$  is the *corpus* probability derived as follows: The smoothed  $n$ -gram probability model  $P(w | h)$  is naturally extended to a sentence  $s = w_0 \dots w_l$ , where  $w_0 = \langle s \rangle$  and  $w_l = \langle /s \rangle$  are the sentence initial and final words, by  $P(s) = \prod_{i=1}^l P(w_i | h_i^n)$ . The corpus probability  $s_1 \dots s_r$  is taken as:

$$\hat{P}(s_1 \dots s_r) = (1 - \lambda) \lambda^{r-1} \prod_{i=1}^r P(s_i) \quad (10)$$

where  $\lambda$  parameterizes the corpus length distribution.<sup>2</sup> Assuming the n-gram language model automaton  $G$  has a single final state  $\langle /s \rangle$  into

<sup>2</sup> $\hat{P}$  models words in a corpus rather than a single sentence since Equation 9 tends to zero as  $m \rightarrow \infty$  otherwise. In Markov chain terms, the corpus distribution is made irreducible to allow a non-trivial stationary distribution.

which all  $\langle /s \rangle$  arcs enter, adding a  $\lambda$  weighted  $\epsilon$  arc from the  $\langle /s \rangle$  state to the initial state and having a final weight  $1 - \lambda$  in order to leave the automaton at the  $\langle /s \rangle$  state will model this corpus distribution. According to Eq. 9,  $P(h)$  is then the stationary distribution of the finite irreducible Markov Chain defined by this altered automaton. There are many methods for computing such a stationary distribution; we use the well-known power method (Stewart, 1999).

One difficulty remains to be resolved. The backoff arcs have a special interpretation in the automaton: they are traversed only if a word fails to match at the higher order. These failure arcs must be properly handled before applying standard stationary distribution calculations. A simple approach would be for each word  $w'$  and state  $h$  such that  $hw' \notin G$ , but  $h'w' \in G$ , add a  $w'$  arc from state  $h$  to  $h'w'$  with weight  $\alpha(h, h')\beta(h'w')$  and then remove all failure arcs (see Figure 2a). This however results in an automaton with  $|V|$  arcs leaving every state, which is unwieldy with larger vocabularies and n-gram orders. Instead, for each word  $w$  and state  $h$  such that  $hw \in G$ , add a  $w$  arc from state  $h$  to  $h'w$  with weight  $-\alpha(h, h')\beta(h'w)$  and then replace all failure labels with  $\epsilon$  labels (see Figure 2b). In this case, the added negatively-weighted arcs compensate for the excess probability mass allowed by the epsilon arcs<sup>3</sup>. The number of added arcs is no more than found in the original model.

## 4.2 Accumulation of higher order values

We are summing over all possible histories of length  $n$  in equation 8, and the steady state probability calculation outlined in the previous section includes the probability mass for histories  $h \notin G$ . The probability mass of states not in  $G$  ends up being allocated to the state representing their longest suffix that is explicitly in  $G$ . That is the state that would be active when these histories are encountered. Hence, once we have calculated the steady state probabilities for each state in the smoothed model, we only need to sum over states explicitly in the model.

As stated earlier, the use of  $\beta(h^wGw)$  in the numerator of equation 8 for  $h^wG$  that are larger than  $h'$  implies that the longer n-grams must be

<sup>3</sup>Since each negatively-weighted arc leaving a state exactly cancels an epsilon arc followed by a matching positively-weighted arc in each iteration of the power method, convergence is assured.

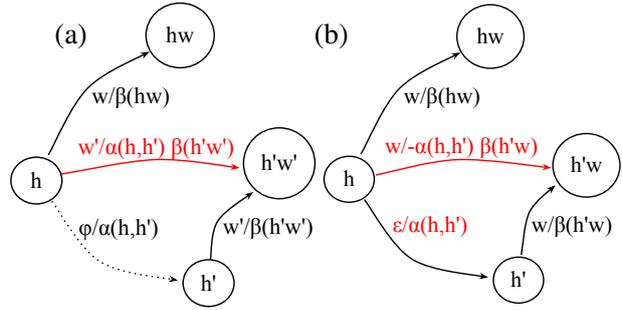


Figure 2: Schemata showing failure arc handling: (a)  $\phi$  removal: add  $w'$  arc (red), delete  $\phi$  arc; (b)  $\phi$  replacement: add  $w$  arc (red), replace  $\phi$  by  $\epsilon$  (red)

re-estimated first. Thus we process each history length in descending order, finishing with the unigram state. Since we assume that, for every n-gram  $hw \in G$ , every prefix and suffix is also in  $G$ , we know that if  $h'w \notin G$  then there is no history  $h$  such that  $h'$  is a suffix of  $h$  and  $hw \in G$ . This allows us to recursively accumulate the  $\alpha(h, h')P(h)$  in the denominator of Eq. 8.

For every n-gram, we can accumulate values required to calculate the three terms in equation 8, and pass them along to calculate lower order n-gram values. Note, however, that a naive implementation of an algorithm to assign these values is  $O(|V|^n)$ . This is due to the fact that the denominator factor must be accumulated for all higher order states that do *not* have the given n-gram. Hence, for every state  $h$  directly backing off to  $h'$  (order  $|V|$ ), and for every n-gram arc leaving state  $h'$  (order  $|V|$ ), some value must be accumulated. This can be particularly clearly seen at the unigram state, which has an arc for every unigram (the size of the vocabulary): for every bigram state (also order of the vocabulary), in the naive algorithm we must look for every possible arc. Since there are  $O(|V|^{n-2})$  lower order histories in the model in the worst case, we have overall complexity  $O(|V|^n)$ . However, we know that the number of stored n-grams is very sparse relative to the possible number of n-grams, so the typical case complexity is far lower. Importantly, the denominator is calculated by first assuming that *all* higher order states back off to the current n-gram, then subtract out the mass associated with those that are already observed at the higher order. In such a way, we need only perform work for higher order n-grams  $hw$  that are explicitly in the model. This optimization achieves orders-of-magnitude speedups, so that models take seconds to process.

Because smoothing is not necessarily con-

strained across n-gram orders, it is possible that higher-order n-grams could be smoothed less than lower order n-grams, so that the numerator of equation 8 can be less than zero, which is not valid. A value less than zero means that the higher order n-grams will already produce the n-gram more frequently than its smoothed expected frequency. We set a minimum value  $\epsilon$  for the numerator, and any n-gram numerator value less than  $\epsilon$  is replaced with  $\epsilon$  (for the current study,  $\epsilon = 0.001$ ). We find this to be relatively infrequent, about 1% of n-grams for most models.

### 4.3 Iteration

Recall that  $\bar{P}$  and  $\beta$  terms on the right-hand side of equation 8 are given and do not change. But there are two other terms in the equation that change as we update the n-gram parameters. The  $\alpha(h, h')$  backoff weights in the denominator ensure normalization at the higher order states, and change as the n-gram parameters at the current state are modified. Further, the steady state probabilities will change as the model changes. Hence, at each state, we must iterate the calculation of the denominator term: first adjust n-gram weights and normalize; then recalculate backoff weights at higher order states and iterate. Since this only involves the denominator term, each n-gram weight can be updated by multiplying by the ratio of the old term and the new term.

After the entire model has been re-estimated, the steady state probability calculation presented in Section 4.1 is run again and model estimation happens again. As we shall see in the experimental results, this typically converges after just a few iterations. At this time, we have no convergence proofs for either of these iterative components to the algorithm, but expect that something can be said about this, which will be a priority in future work.

## 5 Experimental results

All results presented here are for English Broadcast News. We received scripts for replicating the Chelba et al. (2010) results from the authors, and we report statistics on our replication of their paper’s results in Table 2. The scripts are distributed in such a way that the user supplies the data from LDC98T31 (1996 CSR HUB4 Language Model corpus) and the script breaks the collection into training and testing sets, normalizes the text, and

Smoothing method	Perplexity		n-grams ( $\times 1000$ )	
	full	pruned	model	diff
Abs.Disc.	120.4	197.1	382.3	-1.1
Witten-Bell	118.7	196.1	379.3	-1.1
Ristad	126.2	203.4	394.6	-1.1
Katz	119.7	197.9	385.1	-1.1
Kneser-Ney <sup>†</sup>	114.4	234.1	375.4	-12.7

Table 2: Replication of Chelba et al. (2010) using provided script. Using the script, the size of the unpruned model is 31,091,219 ngrams, 4,041 fewer than Chelba et al. (2010). <sup>†</sup> Kneser-Ney model pruned using `-prune-history-lm` switch in SRILM.

trains and prunes the language models using the SRILM toolkit (Stolcke et al., 2011). Presumably due to minor differences in text normalization, resulting in very slightly fewer n-grams in all conditions, we achieve negligibly lower perplexities (one or two tenths of a point) in all conditions, as can be seen when comparing with Table 1. All of the same trends result, thus that paper’s result is successfully replicated here. Note that we ran our Kneser-Ney pruning (noted with a <sup>†</sup> in the table), using the new `-prune-history-lm` switch in SRILM – created in response to the Chelba et al. (2010) paper – which allows the use of another model to calculate the state marginals for pruning. This fixes part of the problem – perplexity does not degrade as much as the Kneser-Ney pruned model in Table 1 – but, as argued earlier in this paper, this is not the sole reason for the degradation and the perplexity remains extremely inflated.

We follow Chelba et al. (2010) in training and test set definition, vocabulary size, and parameters for reporting perplexity. Note that unigrams in the models are never pruned, hence all models assign probabilities over an identical vocabulary and perplexity is comparable across models. For all results reported here, we use the SRILM toolkit for baseline model training and pruning, then convert from the resulting ARPA format model to an OpenFst format (Allauzen et al., 2007), as used in the OpenGrm n-gram library (Roark et al., 2012). We then apply the marginal distribution constraints, and convert the result back to ARPA format for perplexity evaluation with the SRILM toolkit. All models are subjected to full normalization sanity checks, as with typical model functions in the OpenGrm library.

Recall that our algorithm assumes that, for every n-gram in the model, all prefix and suffix n-grams are also in the model. For pruned models, the SRILM toolkit does not impose such a requirement, hence explicit arcs are added to the

Smoothing Method	Perplexity			n-grams ( $\times 1000$ ) in WFST
	Pruned Model	Pruned +MDC	$\Delta$	
Abs.Disc.	197.1	187.4	9.7	389.2
Witten-Bell	196.1	185.7	10.4	385.0
Ristad	203.4	190.3	13.1	395.9
Katz	197.9	187.5	10.4	390.8
AD, WB, Katz Mixture	196.6	186.3	10.3	388.7

Table 3: Perplexity reductions achieved with marginal distribution constraints (MDC) on the heavily pruned models from Chelba et al. (2010), and a mixture model. WFST n-gram counts are slightly higher than ARPA format in Table 2 due to adding prefix and suffix n-grams.

model during conversion, with probability equal to what they would receive in the the original model. The resulting model is equivalent, but with a small number of additional arcs in the explicit representation (around 1% for the most heavily pruned models).

Table 3 presents perplexity results for models that result from applying our marginal distribution constraints to the four heavily pruned models from Table 2. In all four cases, we get perplexity reductions of around 10 points. We present the number of n-grams represented explicitly in the WFST, which is a slight increase from those presented in Table 2 due to the reintroduction of prefix and suffix n-grams.

In addition to the four models reported in Chelba et al. (2010), we produced a mixture model by interpolating (with equal weight) smoothed n-gram probabilities from the full (unpruned) absolute discounting, Witten-Bell and Katz models, which share the same set of n-grams. After renormalizing and pruning to approximately the same size as the other models, we get commensurate gains using this model as with the other models.

Figure 3 demonstrates the importance of iterating the steady state history calculation. All of the methods achieve perplexity reductions with subsequent iterations. Katz and absolute discounting achieve very little reduction in the first iteration, but catch back up in the second iteration.

The other iterative part of the algorithm, discussed in Section 4.3, is the denominator of equation 8, which changes due to adjustments in the backoff weights required by the revised n-gram probabilities. If we do not iteratively update the backoff weights when reestimating the weights, the ‘Pruned+MDC’ perplexities in Table 3 increase by between 0.2–0.4 points. Hence, iterating the steady state probability calculation is quite important, as illustrated by Figure 3; iterating the

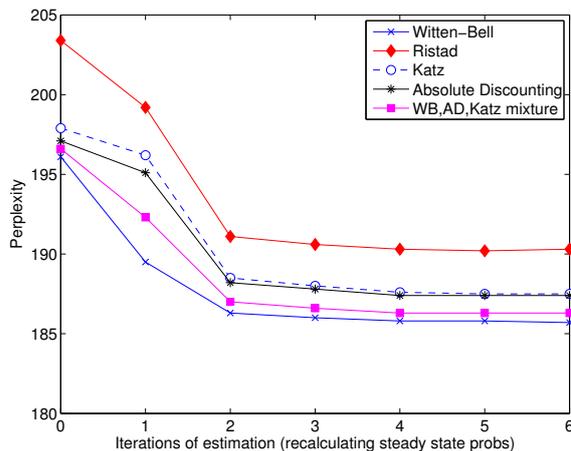


Figure 3: Models resulting from different numbers of parameter re-estimation iterations. Iteration 0 is the baseline pruned model.

denominator calculation much less so, at least for these models. We noted in Section 3 that a key difference between our approach and Kneser and Ney (1995) is that their approach treated the denominator as a constant. If we do this, the ‘Pruned+MDC’ perplexities increase by between 4.5–5.6 points, i.e., about half of the perplexity reduction is lost for each method. Thus, while iteration of denominator calculation may not be critical, it should not be treated as a constant.

We now look at the impacts on system performance we can achieve with these new models<sup>4</sup>, and whether the perplexity differences that we observe translate to real error rate reductions.

For automatic speech recognition experiments, we used as test set the 1997 Hub4 evaluation set consisting of 32,689 words. The acoustic model is a tied-state triphone GMM-based HMM whose input features are 9-frame stacked 13-dimensional PLP-cepstral coefficients projected down to 39 dimensions using LDA. The model was trained on the 1996 and 1997 Hub4 acoustic model training sets (about 150 hours of data) using semi-tied covariance modeling and CMLLR-based speaker adaptive training and 4 iterations of boosted MMI.

We used a multi-pass decoding strategy: two quick passes for adaptation supervision, CMLLR and MLLR estimation; then a slower full decoding pass running about 3 times slower than real time.

Table 4 presents recognition results for the heavily pruned models that we have been considering, both for first pass decoding and rescoreing of the resulting lattices using failure transitions rather than epsilon backoff approximations.

<sup>4</sup>For space purposes, we exclude the Ristad method from this point forward since it is not competitive with the others.

Smoothing Method	Word error rate (WER)			
	First pass		Rescoring	
	Pruned Model	Pruned +MDC	Pruned Model	Pruned +MDC
Abs.Disc.	20.5	19.7	20.2	19.6
Witten-Bell	20.5	19.9	20.1	19.6
Katz	20.5	19.7	20.2	19.7
Mixture	20.5	19.6	20.2	19.6
Kneser-Ney <sup>a</sup>	22.1		22.2	
Kneser-Ney <sup>b</sup>	20.5		20.6	

Table 4: WER reductions achieved with marginal distribution constraints (MDC) on the heavily pruned models from Chelba et al. (2010), and a mixture model. Kneser-Ney results are shown for: a) original pruning; and b) with `-prune-history-lm` switch.

The perplexity reductions that were achieved for these models do translate to real word error rate reductions at both stages of between 0.5 and 0.9 percent absolute. All of these gains are statistically significant at  $p < 0.0001$  using the stratified shuffling test (Yeh, 2000). For pruned Kneser-Ney models, fixing the state marginals with the `-prune-history-lm` switch reduces the WER versus the original pruned model, but no reductions were achieved vs. baseline methods.

Table 5 presents perplexity and WER results for less heavily pruned models, where the pruning thresholds were set to yield approximately 1.5 million n-grams (4 times more than the previous models); and another set at around 5 million n-grams, as well as the full, unpruned models. While the robust gains we’ve observed up to now persist with the 1.5M n-gram models (WER reductions significant, Witten-Bell at  $p < 0.02$ , others at  $p < 0.0001$ ), the larger models yield diminishing gains, with no real WER improvements. Performance of Witten-Bell models with the marginal distribution constraints degrade badly for the larger models, indicating that this method of regularization, unmodified by aggressive pruning, does not provide a well suited distribution for

this sort of optimization. We speculate that this is due to underregularization, having noted some floating point precision issues when allowing the backoff recalculation to run indefinitely.

## 6 Summary and Future Directions

The presented method reestimates lower order n-gram model parameters for a given smoothed backoff model, achieving perplexity and WER reductions for many smoothed models. There remain a number of open questions to investigate in the future. Recall that the numerator in Eq. 8 can be less than zero, meaning that no parameterization would lead to a model with the target frequency of the lower order n-gram, presumably due to over- or under-regularization. We anticipate a pre-constraint on the baseline smoothing method, that would recognize this problem and adjust the smoothing to ensure that a solution does exist. Additionally, it is clear that different regularization methods yield different behaviors, notably that large, relatively lightly pruned Witten-Bell models yield poor results. We will look to identify the issues with such models and provide general guidelines for prepping models prior to processing. Finally, we would like to perform extensive controlled experimentation to examine the relative contribution of the various aspects of our approach.

## Acknowledgments

Thanks to Ciprian Chelba and colleagues for the scripts to replicate their results. This work was supported in part by a Google Faculty Research Award and NSF grant #IIS-0964102. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the NSF.

Smoothing Method	M D C	Less heavily pruned model				Moderately pruned model				Full model			
		ngrams ( $\times 10^6$ )	PPL	WER		ngrams ( $\times 10^6$ )	PPL	WER		ngrams ( $\times 10^6$ )	PPL	WER	
				FP	RS			FP	RS			FP	RS
Abs. Disc.	N Y	1.53	146.6	18.1	17.9	5.19	129.1	17.0	16.6	31.1	120.4	16.2	16.1
			141.2	17.2	17.2		126.3	16.6	16.6	31.1	117.0	16.0	16.0
Witten-Bell	N Y	1.54	145.8	18.1	17.6	5.08	129.4	17.3	16.8	31.1	118.7	16.3	16.1
			139.7	17.9	17.4		126.4	18.4	17.3	31.1	118.4	18.1	17.6
Katz	N Y	1.57	146.6	17.8	17.7	5.10	128.9	16.8	16.6	31.1	119.7	16.2	16.1
			141.1	17.3	17.3		125.7	16.6	16.6	31.1	114.7	16.2	16.1
Mixture	N Y	1.55	145.5	18.1	17.7	5.11	128.2	16.9	16.6	31.1	118.5	16.3	16.1
			139.2	17.3	17.2		123.6	16.6	16.4	31.1	114.6	17.3	16.4
Kneser-Ney backoff model, unpruned:										31.1	114.4	15.8	15.9

Table 5: Perplexity (PPL) and both first pass (FP) and rescoring (RS) WER reductions for less heavily pruned models using marginal distribution constraints (MDC).

## References

- Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. OpenFst: A general and efficient weighted finite-state transducer library. In *Proceedings of the Twelfth International Conference on Implementation and Application of Automata (CIAA 2007), Lecture Notes in Computer Science*, volume 4793, pages 11–23.
- Ciprian Chelba, Thorsten Brants, Will Neveitt, and Peng Xu. 2010. Study on interaction between entropy pruning and Kneser-Ney smoothing. In *Proceedings of Interspeech*, page 24222425.
- Stanley Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report, TR-10-98, Harvard University.
- Joshua Goodman. 2001. A bit of progress in language modeling. *Computer Speech and Language*, 15(4):403–434.
- Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustic, Speech, and Signal Processing*, 35(3):400–401.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 181–184.
- Hermann Ney, Ute Essen, and Reinhard Kneser. 1994. On structuring probabilistic dependences in stochastic language modeling. *Computer Speech and Language*, 8:1–38.
- Eric S. Ristad. 1995. A natural law of succession. Technical Report, CS-TR-495-95, Princeton University.
- Brian Roark, Richard Sproat, Cyril Allauzen, Michael Riley, Jeffrey Sorensen, and Terry Tai. 2012. The OpenGrm open-source finite-state grammar software libraries. In *Proceedings of the ACL 2012 System Demonstrations*, pages 61–66.
- Kristie Seymore and Ronald Rosenfeld. 1996. Scalable backoff language models. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*.
- Vesa Siivola, Teemu Hirsimäki, and Sami Virpioja. 2007. On growing and pruning kneserney smoothed n-gram models. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(5):1617–1624.
- William J Stewart. 1999. Numerical methods for computing stationary distributions of finite irreducible markov chains. *Computational Probability*, pages 81–111.
- Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. Srilm at sixteen: Update and outlook. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.
- Andreas Stolcke. 1998. Entropy-based pruning of backoff language models. In *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274.
- David Talbot and Thorsten Brants. 2008. Randomized language models via perfect hash functions. In *Proceedings of ACL-08: HLT*, pages 505–513.
- David Talbot and Miles Osborne. 2007. Smoothed Bloom filter language models: Tera-scale LMs on the cheap. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 468–476.
- Ian H. Witten and Timothy C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094.
- A. Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International COLING*, pages 947–953.

# Grounded Language Learning from Video Described with Sentences

Haonan Yu and Jeffrey Mark Siskind

Purdue University

School of Electrical and Computer Engineering

465 Northwestern Ave.

West Lafayette, IN 47907-2035 USA

haonan@haonanyu.com, qobi@purdue.edu

## Abstract

We present a method that learns representations for word meanings from short video clips paired with sentences. Unlike prior work on learning language from symbolic input, our input consists of video of people interacting with multiple complex objects in outdoor environments. Unlike prior computer-vision approaches that learn from videos with verb labels or images with noun labels, our labels are sentences containing nouns, verbs, prepositions, adjectives, and adverbs. The correspondence between words and concepts in the video is learned in an unsupervised fashion, even when the video depicts simultaneous events described by multiple sentences or when different aspects of a single event are described with multiple sentences. The learned word meanings can be subsequently used to automatically generate description of new video.

## 1 Introduction

People learn language through exposure to a rich perceptual context. Language is grounded by mapping words, phrases, and sentences to meaning representations referring to the world. Siskind (1996) has shown that even with referential uncertainty and noise, a system based on *cross-situational* learning can robustly acquire a lexicon, mapping words to word-level meanings from sentences paired with sentence-level meanings. However, it did so only for symbolic representations of word- and sentence-level meanings that were not perceptually grounded. An ideal system would not require detailed word-level labelings to acquire word meanings from video but rather could learn language in a largely unsupervised fashion, just as a child does, from video paired with sentences.

There has been recent research on grounded language learning. Roy (2002) pairs training sentences with vectors of real-valued features extracted from synthesized images which depict 2D blocks-world scenes, to learn a specific set of features for adjectives, nouns, and adjuncts. Yu and Ballard (2004) paired training images containing multiple objects with spoken name candidates for the objects to find the correspondence between lexical items and visual features. Dominey and Boucher (2005) paired narrated sentences with symbolic representations of their meanings, automatically extracted from video, to learn object names, spatial-relation terms, and event names as a mapping from the grammatical structure of a sentence to the semantic structure of the associated meaning representation. Chen and Mooney (2008) learned the language of sportscasting by determining the mapping between game commentaries and the meaning representations output by a rule-based simulation of the game. Kwiatkowski et al. (2012) present an approach that learns Montague-grammar representations of word meanings together with a combinatory categorial grammar (CCG) from child-directed sentences paired with first-order formulas that represent their meaning.

Although most of these methods succeed in learning word meanings from sentential descriptions they do so only for symbolic or simple visual input (often synthesized); they fail to bridge the gap between language and computer vision, *i.e.*, they do not attempt to extract meaning representations from complex visual scenes. On the other hand, there has been research on training object and event models from large corpora of complex images and video in the computer-vision community (Kuznetsova et al., 2012; Sadanand and Corso, 2012; Kulkarni et al., 2011; Ordonez et al., 2011; Yao et al., 2010). However, most such work requires training data that labels individual concepts with individual words (*i.e.*, ob-

jects delineated via bounding boxes in images as nouns and events that occur in short video clips as verbs). There is no attempt to model phrasal or sentential meaning, let alone acquire the object or event models from training data labeled with phrasal or sentential annotations. Moreover, such work uses distinct representations for different parts of speech; *i.e.*, object and event recognizers use different representations.

In this paper, we present a method that learns representations for word meanings from short video clips paired with sentences. Our work differs from prior work in three ways. First, our input consists of realistic video filmed in an outdoor environment. Second, we learn the entire lexicon, including nouns, verbs, prepositions, adjectives, and adverbs, simultaneously from video described with whole sentences. Third we adopt a uniform representation for the meanings of words in all parts of speech, namely Hidden Markov Models (HMMs) whose states and distributions allow for multiple possible interpretations of a word or a sentence in an ambiguous perceptual context.

We employ the following representation to ground the meanings of words, phrases, and sentences in video clips. We first run an object detector on each video frame to yield a set of *detections*, each a subregion of the frame. In principle, the object detector need just detect the objects rather than classify them. In practice, we employ a collection of class-, shape-, pose-, and viewpoint-specific detectors and pool the detections to account for objects whose shape, pose, and viewpoint may vary over time. Our methods can learn to associate a single noun with detections produced by multiple detectors. We then string together detections from individual frames to yield *tracks* for objects that temporally span the video clip. We associate a feature vector with each frame (detection) of each such track. This feature vector can encode image features (including the identity of the particular detector that produced that detection) that correlate with object class; region color, shape, and size features that correlate with object properties; and motion features, such as linear and angular object position, velocity, and acceleration, that correlate with event properties. We also compute features between pairs of tracks to encode the relative position and motion of the pairs of objects that participate in events that involve two participants. In principle, we can also compute features

between tuples of any number of tracks.

Following Yamoto et al. (1992), Siskind and Morris (1996), and Starner et al. (1998), we represent the meaning of an intransitive verb, like *jump*, as a two-state HMM over the velocity-direction feature, modeling the requirement that the participant move upward then downward. We represent the meaning of a transitive verb, like *pick up*, as a two-state HMM over both single-object and object-pair features: the agent moving toward the patient while the patient is at rest, followed by the agent moving together with the patient. We extend this general approach to other parts of speech. Nouns, like *person*, can be represented as one-state HMMs over image features that correlate with the object classes denoted by those nouns. Adjectives, like *red*, *round*, and *big*, can be represented as one-state HMMs over region color, shape, and size features that correlate with object properties denoted by such adjectives. Adverbs, like *quickly*, can be represented as one-state HMMs over object-velocity features. Intransitive prepositions, like *leftward*, can be represented as one-state HMMs over velocity-direction features. Static transitive prepositions, like *to the left of*, can be represented as one-state HMMs over the relative position of a pair of objects. Dynamic transitive prepositions, like *towards*, can be represented as HMMs over the changing distance between a pair of objects. Note that with this formulation, the representation of a verb, like *approach*, might be the same as a dynamic transitive preposition, like *towards*. While it might seem like overkill to represent the meanings of words as one-state-HMMs, in practice, we often instead encode such concepts with multiple states to allow for temporal variation in the associated features due to changing pose and viewpoint as well as deal with noise and occlusion. Moreover, the general framework of modeling word meanings as temporally variant time series via multi-state HMMs allows one to model denominalized verbs, *i.e.*, nouns that denote events, as in *The jump was fast*.

Our HMMs are parameterized with varying arity. Some, like  $jump(\alpha)$ ,  $person(\alpha)$ ,  $red(\alpha)$ ,  $round(\alpha)$ ,  $big(\alpha)$ ,  $quickly(\alpha)$ , and  $leftward(\alpha)$  have one argument, while others, like  $pick-up(\alpha, \beta)$ ,  $to-the-left-of(\alpha, \beta)$ , and  $towards(\alpha, \beta)$ , have two arguments (In principle, any arity can be supported.). HMMs are instantiated by mapping their arguments to tracks. This

involves computing the associated feature vector for that HMM over the detections in the tracks chosen to fill its arguments. This is done with a two-step process to support compositional semantics. The meaning of a multi-word phrase or sentence is represented as a joint likelihood of the HMMs for the words in that phrase or sentence. Compositionality is handled by *linking* or *coindexing* the arguments of the conjoined HMMs. Thus a sentence like *The person to the left of the backpack approached the trash-can* would be represented as a conjunction of  $person(p_0)$ ,  $to\text{-}the\text{-}left\text{-}of(p_0, p_1)$ ,  $backpack(p_1)$ ,  $approached(p_0, p_2)$ , and  $trash\text{-}can(p_2)$  over the three participants  $p_0$ ,  $p_1$ , and  $p_2$ . This whole sentence is then grounded in a particular video by mapping these participants to particular tracks and instantiating the associated HMMs over those tracks, by computing the feature vectors for each HMM from the tracks chosen to fill its arguments.

Our algorithm makes six assumptions. First, we assume that we know the part of speech  $C_m$  associated with each lexical entry  $m$ , along with the part-of-speech dependent number of states  $I_c$  in the HMMs used to represent word meanings in that part of speech, the part-of-speech dependent number of features  $N_c$  in the feature vectors used by HMMs to represent word meanings in that part of speech, and the part-of-speech dependent feature-vector computation  $\Phi_c$  used to compute the features used by HMMs to represent word meanings in that part of speech. Second, we pair individual sentences each with a short video clip that depicts that sentence. The algorithm is not able to determine the alignment between multiple sentences and longer video segments. Note that there is no requirement that the video depict *only* that sentence. Other objects may be present and other events may occur. In fact, nothing precludes a training corpus with multiple copies of the same video, each paired with a different sentence describing a different aspect of that video. Moreover, our algorithm potentially can handle a small amount of noise, where a video clip is paired with an incorrect sentence that the video does not depict. Third, we assume that we already have (pre-trained) low-level object detectors capable of detecting instances of our target event participants in individual frames of the video. We allow such detections to be unreliable; our method can handle a moderate amount of false positives

and false negatives. We do not need to know the mapping from these object-detection classes to words; our algorithm determines that. Fourth, we assume that we know the arity of each word in the corpus, *i.e.*, the number of arguments that that word takes. For example, we assume that we know that the word  $person(\alpha)$  takes one argument and the word  $approached(\alpha, \beta)$  takes two arguments. Fifth, we assume that we know the total number of distinct participants that collectively fill all of the arguments for all of the words in each training sentence. For example, for the sentence *The person to the left of the backpack approached the trash-can*, we assume that we know that there are three distinct objects that participate in the event denoted. Sixth, we assume that we know the *argument-to-participant mapping* for each training sentence. Thus, for example, for the above sentence we would know  $person(p_0)$ ,  $to\text{-}the\text{-}left\text{-}of(p_0, p_1)$ ,  $backpack(p_1)$ ,  $approached(p_0, p_2)$ , and  $trash\text{-}can(p_2)$ . The latter two items can be determined by parsing the sentence, which is what we do. One can imagine learning the ability to automatically perform the latter two items, and even the fourth item above, by learning the grammar and the part of speech of each word, such as done by Kwiatkowski et al. (2012). We leave such for future work.

Figure 1 illustrates a single frame from a potential training sample provided as input to our learner. It consists of a video clip paired with a sentence, where the arguments of the words in the sentence are mapped to participants. From a sequence of such training samples, our learner determines the objects tracks and the mapping from participants to those tracks, together with the meanings of the words.

The remainder of the paper is organized as follows. Section 2 generally describes our problem of lexical acquisition from video. Section 3 introduces our work on the sentence tracker, a method for jointly tracking the motion of multiple objects in a video that participate in a sentimentally-specified event. Section 4 elaborates on the details of our problem formulation in the context of this sentence tracker. Section 5 describes how to generalize and extend the sentence tracker so that it can be used to support lexical acquisition. We demonstrate this lexical acquisition algorithm on a small example in Section 6. Finally, we conclude with a discussion in Section 7.

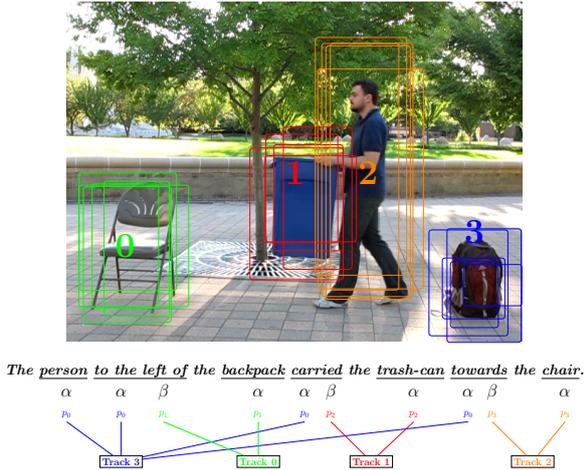


Figure 1: An illustration of our problem. Each word in the sentence has one or more arguments ( $\alpha$  and possibly  $\beta$ ), each argument of each word is assigned to a participant ( $p_0, \dots, p_3$ ) in the event described by the sentence, and each participant can be assigned to any object track in the video. This figure shows a possible (but erroneous) interpretation of the sentence where the mapping is:  $p_0 \mapsto$  **Track 3**,  $p_1 \mapsto$  **Track 0**,  $p_2 \mapsto$  **Track 1**, and  $p_3 \mapsto$  **Track 2**, which might (incorrectly) lead the learner to conclude that the word *person* maps to the backpack, the word *backpack* maps to the chair, the word *trash-can* maps to the trash-can, and the word *chair* maps to the person.

## 2 General Problem Formulation

Throughout this paper, lowercase letters are used for variables or hidden quantities while uppercase ones are used for constants or observed quantities.

We are given a lexicon  $\{1, \dots, M\}$ , letting  $m$  denote a lexical entry. We are given a sequence  $D = (D_1, \dots, D_R)$  of video clips  $D_r$ , each paired with a sentence  $S_r$  from a sequence  $S = (S_1, \dots, S_R)$  of sentences. We refer to  $D_r$  paired with  $S_r$  as a *training sample*. Each sentence  $S_r$  is a sequence  $(S_{r,1}, \dots, S_{r,L_r})$  of words  $S_{r,l}$ , each an entry from the lexicon. A given entry may potentially appear in multiple sentences and even multiple times in a given sentence. For example, the third word in the first sentence might be the same entry as the second word in the fourth sentence, in which case  $S_{1,3} = S_{4,2}$ . This is what allows cross-situational learning in our algorithm.

Let us assume, for a moment, that we can process each video clip  $D_r$  to yield a sequence  $(\tau_{r,1}, \dots, \tau_{r,U_r})$  of object tracks  $\tau_{r,u}$ . Let us also assume that  $D_r$  is paired with a sen-

tence  $S_r =$  *The person approached the chair*, specified to have two participants,  $p_{r,0}$  and  $p_{r,1}$ , with the mapping *person*( $p_{r,0}$ ), *chair*( $p_{r,1}$ ), and *approached*( $p_{r,0}, p_{r,1}$ ). Let us further assume, for a moment, that we are given a mapping from participants to object tracks, say  $p_{r,0} \mapsto \tau_{r,39}$  and  $p_{r,1} \mapsto \tau_{r,51}$ . This would allow us to instantiate the HMMs with object tracks for a given video clip: *person*( $\tau_{r,39}$ ), *chair*( $\tau_{r,51}$ ), and *approached*( $\tau_{r,39}, \tau_{r,51}$ ). Let us further assume that we can score each such instantiated HMM and aggregate the scores for all of the words in a sentence to yield a sentence score and further aggregate the scores for all of the sentences in the corpus to yield a corpus score. However, we don't know the parameters of the HMMs. These constitute the unknown meanings of the words in our corpus which we wish to learn. The problem is to simultaneously determine (a) those parameters along with (b) the object tracks and (c) the mapping from participants to object tracks. We do this by finding (a)–(c) that maximizes the corpus score.

## 3 The Sentence Tracker

Barbu et al. (2012a) presented a method that first determines object tracks from a single video clip and then uses these fixed tracks with HMMs to recognize actions corresponding to verbs and construct sentential descriptions with templates. Later Barbu et al. (2012b) addressed the problem of solving (b) and (c), for a single object track constrained by a single intransitive verb, without solving (a), in the context of a single video clip. Our group has generalized this work to yield an algorithm called the *sentence tracker* which operates by way of a factorial HMM framework. We introduce that here as the foundation of our extension.

Each video clip  $D_r$  contains  $T_r$  frames. We run an object detector on each frame to yield a set  $D_r^t$  of detections. Since our object detector is unreliable, we bias it to have high recall but low precision, yielding multiple detections in each frame. We form an object track by selecting a single detection for that track for each frame. For a moment, let us consider a single video clip with length  $T$ , with detections  $D^t$  in frame  $t$ . Further, let us assume that we seek a single object track in that video clip. Let  $j^t$  denote the index of the detection from  $D^t$  in frame  $t$  that is selected to form the track. The object detector scores each detection. Let  $F(D^t, j^t)$  denote that score. More-

over, we wish the track to be temporally coherent; we want the objects in a track to move smoothly over time and not jump around the field of view. Let  $G(D^{t-1}, j^{t-1}, D^t, j^t)$  denote some measure of coherence between two detections in adjacent frames. (One possible such measure is consistency of the displacement of  $D^t$  relative to  $D^{t-1}$  with the velocity of  $D^{t-1}$  computed from the image by optical flow.) One can select the detections to yield a track that maximizes both the aggregate detection score and the aggregate temporal coherence score.

$$\max_{j^1, \dots, j^T} \left( \begin{array}{l} \sum_{t=1}^T F(D^t, j^t) \\ + \sum_{t=2}^T G(D^{t-1}, j^{t-1}, D^t, j^t) \end{array} \right) \quad (1)$$

This can be determined with the Viterbi (1967) algorithm and is known as *detection-based tracking* (Viterbi, 1971).

Recall that we model the meaning of an intransitive verb as an HMM over a time series of features extracted for its participant in each frame. Let  $\lambda$  denote the parameters of this HMM,  $(q^1, \dots, q^T)$  denote the sequence of states  $q^t$  that leads to an observed track,  $B(D^t, j^t, q^t, \lambda)$  denote the conditional log probability of observing the feature vector associated with the detection selected by  $j^t$  among the detections  $D^t$  in frame  $t$ , given that the HMM is in state  $q^t$ , and  $A(q^{t-1}, q^t, \lambda)$  denote the log transition probability of the HMM. For a given track  $(j^1, \dots, j^T)$ , the state sequence that yields the maximal likelihood is given by:

$$\max_{q^1, \dots, q^T} \left( \begin{array}{l} \sum_{t=1}^T B(D^t, j^t, q^t, \lambda) \\ + \sum_{t=2}^T A(q^{t-1}, q^t, \lambda) \end{array} \right) \quad (2)$$

which can also be found by the Viterbi algorithm.

A given video clip may depict multiple objects, each moving along its own trajectory. There may be both a person jumping and a ball rolling. How are we to select one track over the other? The key insight of the sentence tracker is to bias the selection of a track so that it matches an HMM. This is done by combining the cost function of Eq. 1 with the cost function of Eq. 2 to yield Eq. 3, which can also be determined using the Viterbi algorithm. This is done by forming the cross product of the

two lattices. This jointly selects the optimal detections to form the track, together with the optimal state sequence, and scores that combination.

$$\max_{\substack{j^1, \dots, j^T \\ q^1, \dots, q^T}} \left( \begin{array}{l} \sum_{t=1}^T F(D^t, j^t) \\ + B(D^t, j^t, q^t, \lambda) \\ + \sum_{t=2}^T G(D^{t-1}, j^{t-1}, D^t, j^t) \\ + A(q^{t-1}, q^t, \lambda) \end{array} \right) \quad (3)$$

While we formulated the above around a single track and a word that contains a single participant, it is straightforward to extend this so that it supports multiple tracks and words of higher arity by forming a larger cross product. When doing so, we generalize  $j^t$  to denote a sequence of detections from  $D^t$ , one for each of the tracks. We further need to generalize  $F$  so that it computes the joint score of a sequence of detections, one for each track,  $G$  so that it computes the joint measure of coherence between a sequence of pairs of detections in two adjacent frames, and  $B$  so that it computes the joint conditional log probability of observing the feature vectors associated with the sequence of detections selected by  $j^t$ . When doing this, note that Eqs. 1 and 3 maximize over  $j^1, \dots, j^T$  which denotes  $T$  sequences of detection indices, rather than  $T$  individual indices.

It is further straightforward to extend the above to support a sequence  $(S_1, \dots, S_L)$  of words  $S_l$  denoting a sentence, each of which applies to different subsets of the multiple tracks, again by forming a larger cross product. When doing so, we generalize  $q^t$  to denote a sequence  $(q_1^t, \dots, q_L^t)$  of states  $q_l^t$ , one for each word  $l$  in the sentence, and use  $q_l$  to denote the sequence  $(q_l^1, \dots, q_l^T)$  and  $q$  to denote the sequence  $(q_1, \dots, q_L)$ . We further need to generalize  $B$  so that it computes the joint conditional log probability of observing the feature vectors for the detections in the tracks that are assigned to the arguments of the HMM for each word in the sentence and  $A$  so that it computes the joint log transition probability for the HMMs for all words in the sentence. This allows selection of an optimal sequence of tracks that yields the highest score for the sentential meaning of a sequence of words. Modeling the meaning of a sentence through a sequence of words whose meanings are modeled by HMMs, defines a *factorial* HMM for that sentence, since the overall Markov process for that sentence can be factored into inde-

pendent component processes (Brand et al., 1997; Zhong and Ghosh, 2001) for the individual words. In this view,  $q$  denotes the state sequence for the combined factorial HMM and  $q_l$  denotes the factor of that state sequence for word  $l$ . The remainder of this paper wraps this sentence tracker in Baum Welch (Baum et al., 1970; Baum, 1972).

#### 4 Detailed Problem Formulation

We adapt the sentence tracker to training a corpus of  $R$  video clips, each paired with a sentence. Thus we augment our notation, generalizing  $j^t$  to  $j_r^t$  and  $q_l^t$  to  $q_{r,l}^t$ . Below, we use  $j_r$  to denote  $(j_r^1, \dots, j_r^{T_r})$ ,  $j$  to denote  $(j_1, \dots, j_R)$ ,  $q_{r,l}$  to denote  $(q_{r,l}^1, \dots, q_{r,l}^{T_r})$ ,  $q_r$  to denote  $(q_{r,1}, \dots, q_{r,L_r})$ , and  $q$  to denote  $(q_1, \dots, q_R)$ .

We use discrete features, namely natural numbers, in our feature vectors, quantized by a binning process. We assume the part of speech of entry  $m$  is known as  $C_m$ . The length of the feature vector may vary across parts of speech. Let  $N_c$  denote the length of the feature vector for part of speech  $c$ ,  $x_{r,l}$  denote the time-series  $(x_{r,l}^1, \dots, x_{r,l}^{T_r})$  of feature vectors  $x_{r,l}^t$ , associated with  $S_{r,l}$  (which recall is some entry  $m$ ), and  $x_r$  denote the sequence  $(x_{r,1}, \dots, x_{r,L_r})$ . We assume that we are given a function  $\Phi_c(D_r^t, j_r^t)$  that computes the feature vector  $x_{r,l}^t$  for the word  $S_{r,l}$  whose part of speech is  $C_{S_{r,l}} = c$ . Note that we allow  $\Phi$  to be dependent on  $c$  allowing different features to be computed for different parts of speech, since we can determine  $m$  and thus  $C_m$  from  $S_{r,l}$ . We choose to have  $N_c$  and  $\Phi_c$  depend on the part of speech  $c$  and not on the entry  $m$  since doing so would be tantamount to encoding the to-be-learned word meaning in the provided feature-vector computation.

The goal of training is to find a sequence  $\lambda = (\lambda_1, \dots, \lambda_M)$  of parameters  $\lambda_m$  that best explains the  $R$  training samples. The parameters  $\lambda_m$  constitute the meaning of the entry  $m$  in the lexicon. Collectively, these are the initial state probabilities  $a_{0,k}^m$ , for  $1 \leq k \leq I_{C_m}$ , the transition probabilities  $a_{i,k}^m$ , for  $1 \leq i, k \leq I_{C_m}$ , and the output probabilities  $b_{i,n}^m(x)$ , for  $1 \leq i \leq I_{C_m}$  and  $1 \leq n \leq N_{C_m}$ , where  $I_{C_m}$  denotes the number of states in the HMM for entry  $m$ . Like before, we could have a distinct  $I_m$  for each entry  $m$  but instead have  $I_{C_m}$  depend only on the part of speech of entry  $m$ , and assume that we know the fixed  $I$  for each part of speech. In our case,  $b_{i,n}^m$  is a discrete distribution because the features are binned.

## 5 The Learning Algorithm

Instantiating the above approach requires a definition for what it means to *best explain the  $R$  training samples*. Towards this end, we define the score of a video clip  $D_r$  paired with sentence  $S_r$  given the parameter set  $\lambda$  to characterize how well this training sample is explained. While the cost function in Eq. 3 may qualify as a score, it is easier to fit a likelihood calculation into the Baum-Welch framework than a MAP estimate. Thus we replace the max in Eq. 3 with a  $\sum$  and redefine our scoring function as follows:

$$L(D_r; S_r, \lambda) = \sum_{j_r} P(j_r | D_r) P(x_r | S_r, \lambda) \quad (4)$$

The score in Eq. 4 can be interpreted as an expectation of the HMM likelihood over all possible mappings from participants to all possible tracks. By definition,  $P(j_r | D_r) = \frac{V(D_r, j_r)}{\sum_{j_r'} V(D_r, j_r')}$ , where the numerator is the score of a particular track sequence  $j_r$  while the denominator sums the scores over all possible track sequences. The log of the numerator  $V(D_r, j_r)$  is simply Eq. 1 without the max. The log of the denominator can be computed efficiently by the forward algorithm (Baum and Petrie, 1966). The likelihood for a factorial HMM can be computed as:

$$P(x_r | S_r, \lambda) = \sum_{q_r} \prod_l P(x_{r,l}, q_{r,l} | S_{r,l}, \lambda) \quad (5)$$

*i.e.*, summing the likelihoods for all possible state sequences. Each summand is simply the joint likelihood for all the words in the sentence conditioned on a state sequence  $q_r$ . For HMMs we have

$$P(x_{r,l}, q_{r,l} | S_{r,l}, \lambda) = \prod_t a_{q_{r,l}^{t-1}, q_{r,l}^t}^{S_{r,l}} \prod_n b_{q_{r,l}^t, n}^{S_{r,l}}(x_{r,l,n}^t) \quad (6)$$

Finally, for a training corpus of  $R$  samples, we seek to maximize the joint score:

$$L(D; S, \lambda) = \prod_r L(D_r; S_r, \lambda) \quad (7)$$

A local maximum can be found by employing the Baum-Welch algorithm (Baum et al., 1970; Baum, 1972). By constructing an auxiliary function (Bilmes, 1997), one can derive the reestimation formulas in Eq. 8, where  $x_{r,l,n}^t = h$  denotes the selection of all possible  $j_r^t$  such that the  $n$ th

$$\begin{aligned}
a_{i,k}^m &= \theta_i^m \sum_{r=1}^R \sum_{l=1}^{L_r} \sum_{t=1}^{T_r} \underbrace{\frac{L(q_{r,l}^{t-1} = i, q_{r,l}^t = k, D_r; S_r, \lambda')}{L(D_r; S_r, \lambda')}}_{\xi(r,l,i,k,t)} \\
b_{i,n}^m(h) &= \psi_{i,n}^m \sum_{r=1}^R \sum_{l=1}^{L_r} \sum_{t=1}^{T_r} \underbrace{\frac{L(q_{r,l}^t = i, x_{r,l,n}^t = h, D_r; S_r, \lambda')}{L(D_r; S_r, \lambda')}}_{\gamma(r,l,n,i,h,t)}
\end{aligned} \tag{8}$$

feature computed by  $\Phi_{C_m}(D_r^t, j_r^t)$  is  $h$ . The coefficients  $\theta_i^m$  and  $\psi_{i,n}^m$  are for normalization.

The reestimation formulas involve *occurrence counting*. However, since we use a factorial HMM that involves a cross-product lattice and use a scoring function derived from Eq. 3 that incorporates both tracking (Eq. 1) and word models (Eq. 2), we need to count the frequency of transitions in the whole cross-product lattice. As an example of such cross-product occurrence counting, when counting the transitions from state  $i$  to  $k$  for the  $l$ th word from frame  $t - 1$  to  $t$ , i.e.,  $\xi(r, l, i, k, t)$ , we need to count all the possible paths through the adjacent factorial states  $(j_r^{t-1}, q_{r,1}^{t-1}, \dots, q_{r,L_r}^{t-1})$  and  $(j_r^t, q_{r,1}^t, \dots, q_{r,L_r}^t)$  such that  $q_{r,l}^{t-1} = i$  and  $q_{r,l}^t = k$ . Similarly, when counting the frequency of being at state  $i$  while observing  $h$  as the  $n$ th feature in frame  $t$  for the  $l$ th word of entry  $m$ , i.e.,  $\gamma(r, l, n, i, h, t)$ , we need to count all the possible paths through the factorial state  $(j_r^t, q_{r,1}^t, \dots, q_{r,L_r}^t)$  such that  $q_{r,l}^t = i$  and the  $n$ th feature computed by  $\Phi_{C_m}(D_r^t, j_r^t)$  is  $h$ .

The reestimation of a single component HMM can depend on the previous estimate for other component HMMs. This dependence happens because of the argument-to-participant mapping which coindexes arguments of different component HMMs to the same track. **It is precisely this dependence that leads to cross-situational learning of two kinds: both inter-sentential and intra-sentential.** Acquisition of a word meaning is driven across sentences by entries that appear in more than one training sample and within sentences by the requirement that the meanings of all of the individual words in a sentence be consistent with the collective sentential meaning.

## 6 Experiment

We filmed 61 video clips (each 3–5 seconds at 640×480 resolution and 40 fps) that depict a variety of different compound events. Each clip depicts multiple simultaneous events between some

S	→ NP VP
NP	→ D N [PP]
D	→ <i>the</i>
N	→ <i>person</i>   <i>backpack</i>   <i>trash-can</i>   <i>chair</i>
PP	→ P NP
P	→ <i>to the left of</i>   <i>to the right of</i>
VP	→ V NP [ADV] [PPM]
V	→ <i>picked up</i>   <i>put down</i>   <i>carried</i>   <i>approached</i>
ADV	→ <i>quickly</i>   <i>slowly</i>
PPM	→ PM NP
PM	→ <i>towards</i>   <i>away from</i>

Table 1: The grammar used for our annotation and generation. Our lexicon contains 1 determiner, 4 nouns, 2 spatial relation prepositions, 4 verbs, 2 adverbs, and 2 motion prepositions for a total of 15 lexical entries over 6 parts of speech.

subset of four objects: a person, a backpack, a chair, and a trash-can. These clips were filmed in three different outdoor environments which we use for cross validation. We manually annotated each video with several sentences that describe what occurs in that video. The sentences were constrained to conform to the grammar in Table 1. Our corpus of 159 training samples pairs some videos with more than one sentence and some sentences with more than one video, with an average of 2.6 sentences per video <sup>1</sup>.

We model and learn the semantics of all words except determiners. Table 2 specifies the arity, the state number  $I_c$ , and the features computed by  $\Phi_c$  for the semantic models for words of each part of speech  $c$ . While we specify a different subset of features for each part of speech, we presume that, in principle, with enough training data, we could include all features in all parts of speech and automatically learn which ones are noninformative and lead to uniform distributions.

We use an off-the-shelf object detector (Felzenszwalb et al., 2010a; Felzenszwalb et al., 2010b) which outputs detections in the form of scored axis-aligned rectangles. We trained four object detectors, one for each of the four object classes in

<sup>1</sup>Our code, videos, and sentential annotations are available at <http://haonanyu.com/research/acl2013/>.

$c$	arity	$I_c$	$\Phi_c$
N	1	1	$\alpha$ detector index
V	2	3	$\alpha$ VEL MAG
			$\alpha$ VEL ORIENT
			$\beta$ VEL MAG
			$\beta$ VEL ORIENT
P	2	1	$\alpha$ - $\beta$ DIST
			$\alpha$ - $\beta$ size ratio
ADV	1	3	$\alpha$ VEL MAG
PM	2	3	$\alpha$ VEL MAG
			$\alpha$ - $\beta$ DIST

Table 2: Arguments and model configurations for different parts of speech  $c$ . VEL stands for velocity, MAG for magnitude, ORIENT for orientation, and DIST for distance.

our corpus: person, backpack, chair, and trash-can. For each frame, we pick the two highest-scoring detections produced by each object detector and pool the results yielding eight detections per frame. Having a larger pool of detections per frame can better compensate for false negatives in the object detection and potentially yield smoother tracks but it increases the size of the lattice and the concomitant running time and does not lead to appreciably better performance on our corpus.

We compute continuous features, such as velocity, distance, size ratio, and x-position solely from the detection rectangles and quantize the features into bins as follows:

**velocity** To reduce noise, we compute the velocity of a participant by averaging the optical flow in the detection rectangle. The velocity magnitude is quantized into 5 levels: *absolutely stationary*, *stationary*, *moving*, *fast moving*, and *quickly*. The velocity orientation is quantized into 4 directions: *left*, *up*, *right*, and *down*.

**distance** We compute the Euclidean distance between the detection centers of two participants, which is quantized into 3 levels: *near*, *normal*, and *far away*.

**size ratio** We compute the ratio of detection area of the first participant to the detection area of the second participant, quantized into 2 possibilities: *larger/smaller than*.

**x-position** We compute the difference between the x-coordinates of the participants, quantized into 2 possibilities: *to the left/right of*.

The binning process was determined by a preprocessing step that clustered a subset of the training data. We also incorporate the index of the detector that produced the detection as a feature. The par-

ticular features computed for each part of speech are given in Table 2.

Note that while we use English phrases, like *to the left of*, to refer to particular bins of particular features, and we have object detectors which we train on samples of a particular object class such as *backpack*, such phrases are only mnemonic of the clustering and object-detector training process. We do not have a fixed correspondence between the lexical entries and any particular feature value. Moreover, that correspondence need not be one-to-one: a given lexical entry may correspond to a (time variant) constellation of feature values and any given feature value may participate in the meaning of multiple lexical entries.

We perform a three-fold cross validation, taking the test data for each fold to be the videos filmed in a given outdoor environment and the training data for that fold to be all training samples that contain other videos. For testing, we hand selected 24 sentences generated by the grammar in Table 1, where each sentence is true for at least one test video. Half of these sentences (designated NV) contain only nouns and verbs while the other half (designated ALL) contain other parts of speech. The latter are longer and more complicated than the former. We score each testing video paired with every sentence in both NV and ALL. To evaluate our results, we manually annotated the correctness of each such pair.

Video-sentence pairs could be scored with Eq. 4. However, the score depends on the sentence length, the collective numbers of states and features in the HMMs for words in that sentence, and the length of the video clip. To render the scores comparable across such variation we incorporate a sentence prior to the per-frame score:

$$\hat{L}(D_r, S_r; \lambda) = [L(D_r; S_r, \lambda)]^{\frac{1}{|T_r|}} \pi(S_r) \quad (9)$$

where

$$\pi(S_r) = \exp \sum_{l=1}^{L_r} \left( \begin{array}{c} E(I_{C_{S_r,l}}) \\ N_{C_{S_r,l}} \\ + \sum_{n=1} E(Z_{C_{S_r,l},n}) \end{array} \right) \quad (10)$$

In the above,  $Z_{C_{S_r,l},n}$  is the number of bins for the  $n$ th feature of  $S_{r,l}$  of part of speech  $C_{S_r,l}$  and  $E(Y) = -\sum_{y=1}^Y \frac{1}{Y} \log \frac{1}{Y} = \log Y$  is the entropy of a uniform distribution over  $Y$  bins. This prior prefers longer sentences which describe more information in the video.

	CHANCE	BLIND	OUR	HAND
NV	0.155	0.265	0.545	0.748
ALL	0.099	0.198	0.639	0.786

Table 3: F1 scores of different methods.

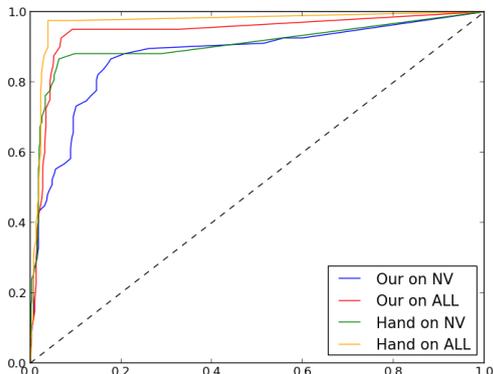


Figure 2: ROC curves of trained models and hand-written models.

The scores are thresholded to decide hits, which together with the manual annotations, can generate TP, TN, FP, and FN counts. We select the threshold that leads to the maximal F1 score on the training set, use this threshold to compute F1 scores on the test set in each fold, and average F1 scores across the folds.

The F1 scores are listed in the column labeled **Our** in Table 3. For comparison, we also report F1 scores for three baselines: **Chance**, **Blind**, and **Hand**. The **Chance** baseline randomly classifies a video-sentence pair as a hit with probability 0.5. The **Blind** baseline determines hits by potentially looking at the sentence but never looking at the video. We can find an upper bound on the F1 score that any blind method could have on each of our test sets by solving a 0-1 fractional programming problem (Dinkelbach, 1967) (see Appendix A for details). The **Hand** baseline determines hits with hand-coded HMMs, carefully designed to yield what we believe is near-optimal performance. As can be seen from Table 3, our trained models perform substantially better than the **Chance** and **Blind** baselines and approach the performance of the ideal **Hand** baseline. One can further see from the ROC curves in Figure 2, comparing the trained and hand-written models on both NV and ALL, that the trained models are close to optimal. Note that performance on ALL exceeds that on NV with the trained models. This is because longer sentences with varied parts of speech incorporate more information into the scoring process.

## 7 Conclusion

We presented a method that learns word meanings from video paired with sentences. Unlike prior work, our method deals with realistic video scenes labeled with whole sentences, not individual words labeling hand delineated objects or events. The experiment shows that it can correctly learn the meaning representations in terms of HMM parameters for our lexical entries, from highly ambiguous training data. Our maximum-likelihood method makes use of only positive sentential labels. As such, it might require more training data for convergence than a method that also makes use of negative training sentences that are not true of a given video. Such can be handled with discriminative training, a topic we plan to address in the future. We believe that this will allow learning larger lexicons from more complex video without excessive amounts of training data.

## Acknowledgments

This research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-10-2-0060. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

## A An Upper Bound on the F1 Score of any Blind Method

A **Blind** algorithm makes identical decisions on the same sentence paired with different video clips. An optimal algorithm will try to find a decision  $s_i$  for each test sentence  $i$  that maximizes the F1 score. Suppose, the ground-truth yields  $FP_i$  false positives and  $TP_i$  true positives on the test set when  $s_i = 1$ . Also suppose that setting  $s_i = 0$  yields  $FN_i$  false negatives. Then the F1 score is

$$F1 = \frac{1}{1 + \underbrace{\frac{\sum_i s_i FP_i + (1 - s_i) FN_i}{\sum_i 2s_i TP_i}}_{\Delta}}$$

Thus we want to minimize the term  $\Delta$ . This is an instance of a 0-1 fractional programming problem which can be solved by binary search or Dinkelbach’s algorithm (Dinkelbach, 1967).

## References

- A. Barbu, A. Bridge, Z. Burchill, D. Coroian, S. Dickinson, S. Fidler, A. Michaux, S. Mussman, N. Siddharth, D. Salvi, L. Schmidt, J. Shangguan, J. M. Siskind, J. Waggoner, S. Wang, J. Wei, Y. Yin, and Z. Zhang. 2012a. Video in sentences out. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, pages 102–112.
- A. Barbu, N. Siddharth, A. Michaux, and J. M. Siskind. 2012b. Simultaneous object detection, tracking, and event recognition. *Advances in Cognitive Systems*, 2:203–220, December.
- L. E. Baum and T. Petrie. 1966. Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37:1554–1563.
- L. E. Baum, T. Petrie, G. Soules, and N. Weiss. 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171.
- L. E. Baum. 1972. An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process. *Inequalities*, 3:1–8.
- J. Bilmes. 1997. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical Report TR-97-021, ICSI.
- M. Brand, N. Oliver, and A. Pentland. 1997. Coupled hidden Markov models for complex action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999.
- D. L. Chen and R. J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of the 25th International Conference on Machine Learning*.
- W. Dinkelbach. 1967. On nonlinear fractional programming. *Management Science*, 13(7):492–498.
- P. F. Dominey and J.-D. Boucher. 2005. Learning to talk about events from narrated video in a construction grammar framework. *Artificial Intelligence*, 167(12):31–61.
- P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. 2010a. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645.
- P. F. Felzenszwalb, R. B. Girshick, and D. A. McAllester. 2010b. Cascade object detection with deformable part models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2241–2248.
- G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. 2011. Baby talk: Understanding and generating simple image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1601–1608.
- P. Kuznetsova, V. Ordonez, A. C. Berg, T. L. Berg, and Y. Choi. 2012. Collective generation of natural image descriptions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pages 359–368.
- T. Kwiatkowski, S. Goldwater, L. Zettlemoyer, and M. Steedman. 2012. A probabilistic model of syntactic and semantic acquisition from child-directed utterances and their meanings. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 234–244.
- V. Ordonez, G. Kulkarni, and T. L. Berg. 2011. Im2text: Describing images using 1 million captioned photographs. In *Proceedings of Neural Information Processing Systems*.
- D. Roy. 2002. Learning visually-grounded words and syntax for a scene description task. *Computer Speech and Language*, 16:353–385.
- S. Sadanand and J. J. Corso. 2012. Action bank: A high-level representation of activity in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1234–1241.
- J. M. Siskind and Q. Morris. 1996. A maximum-likelihood approach to visual event classification. In *Proceedings of the Fourth European Conference on Computer Vision*, pages 347–360.
- J. M. Siskind. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, 61:39–91.
- T. Starner, J. Weaver, and A. Pentland. 1998. Real-time American Sign Language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1371–1375.
- A. J. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–267.
- A. Viterbi. 1971. Convolutional codes and their performance in communication systems. *IEEE Transactions on Communication Technology*, 19(5):751–772.
- J. Yamoto, J. Ohya, and K. Ishii. 1992. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 379–385.

- B. Z. Yao, X. Yang, L. Lin, M. W. Lee, and S.-C. Zhu. 2010. I2T: Image parsing to text description. *Proceedings of the IEEE*, 98(8):1485–1508, August.
- C. Yu and D. H. Ballard. 2004. On the integration of grounding language and learning objects. In *Proceedings of the 19th National Conference on Artificial Intelligence*, pages 488–493.
- S. Zhong and J. Ghosh. 2001. A new formulation of coupled hidden Markov models. Technical report, Department of Electrical and Computer Engineering, The University of Texas at Austin.

# Plurality, Negation, and Quantification: Towards Comprehensive Quantifier Scope Disambiguation

Mehdi Manshadi, Daniel Gildea, and James Allen

University of Rochester  
734 Computer Studies Building  
Rochester, NY 14627  
mehdih, gildea, james@cs.rochester.edu

## Abstract

Recent work on statistical quantifier scope disambiguation (QSD) has improved upon earlier work by scoping an arbitrary number and type of noun phrases. No corpus-based method, however, has yet addressed QSD when incorporating the implicit universal of plurals and/or operators such as negation. In this paper we report early, though promising, results for automatic QSD when handling both phenomena. We also present a general model for learning to build partial orders from a set of pairwise preferences. We give an  $n \log n$  algorithm for finding a guaranteed approximation of the optimal solution, which works very well in practice. Finally, we significantly improve the performance of the previous model using a rich set of automatically generated features.

## 1 Introduction

The sentence *there is one faculty member in every graduate committee* is ambiguous with respect to quantifier scoping, since there are at least two possible readings: If *one* has **wide scope**, there is a unique faculty member on every committee. If *every* has wide scope, there can be different faculty members on each committee. Over the past decade there has been some work on statistical quantifier scope disambiguation (QSD) (Higgins and Sadock, 2003; Galen and MacCartney, 2004; Manshadi and Allen, 2011a). However, the extent of the work has been quite limited for several reasons. First, in the past two decades, the main focus of the NLP community has been on shallow text processing. As a deep processing task, QSD is not essential for many NLP applications that do not require deep understanding. Second, there has been a lack of comprehensive scope-disambiguated corpora, resulting in the lack of work on extensive

statistical QSD. Third, QSD has often been considered only in the context of explicit quantification such as *each* and *every* versus *some* and *a/an*. These co-occurrences do not happen very often in real-life data. For example, Higgins and Sadock (2003) find fewer than 1000 sentences with two or more explicit quantifiers in the Wall Street journal section of Penn Treebank. Furthermore, for more than 60% of those sentences, the order of the quantifiers does not matter, either as a result of the logical equivalence (as in two existentials), or because they do not have any scope interaction.

Having said that, with deep language processing receiving more attention in recent years, QSD is becoming a real-life issue.<sup>1</sup> At the same time, new scope-disambiguated corpora have become available (Manshadi et al., 2011b). In this paper, we aim at tackling the third issue mentioned above. We push statistical QSD beyond explicit quantification, and address an interesting, yet practically important, problem in QSD: plurality and quantification. In spite of an extensive literature in theoretical semantics (Hamm and Hinrichs, 2010; Landmann, 2000), this topic has not been well investigated in computational linguistics. To illustrate the phenomenon, consider (1):

1. *Three words start with a capital letter.*

A deep understanding of this sentence, requires deciding whether each *word* in the set, referred to by *Three words*, starts with a potentially distinct *capital letter* (as in *Apple, Orange, Banana*) or there is a unique *capital letter* which each *word* starts with (as in *Apple, Adam, Athens*). By treating the NP *Three words* as a single atomic entity, earlier work on automatic QSD has overlooked this problem. In general, every plural NP potentially introduces an **implicit universal**, ranging

<sup>1</sup>For example, Liang et al. (2011) in their state-of-the-art statistical semantic parser within the domain of natural language queries to databases, explicitly devise quantifier scoping in the semantic model.

over the collection of entities introduced by the plural.<sup>2</sup> Scoping this implicit universal is just as important. While explicit universals may not occur very often in natural language, the usage of plurals is very common. Plurals form 18% of the NPs in our corpus and 20% of the nouns in Penn Treebank. Explicit universals, on the other hand, form less than 1% of the determiners in Penn Treebank. Quantifiers are also affected by negation. Previous work (e.g., Morante and Blanco, 2012) has investigated automatically detecting the scope and focus of negation. However, the scope of negation with respect to quantifiers is a different phenomenon. Consider the following sentence.

2. *The word does not start with a capital letter.*

Transforming this sentence into a meaning representation language, for almost any practical purposes, requires deciding whether the NP *a capital letter* lies in the scope of the negation or outside of it. The former describes the preferred reading where *The word* starts with a lowercase letter as in *apple, orange, banana*, but the latter gives the unlikely reading, according to which there exists a particular *capital letter*, say *A*, that *The word* starts with, as in *apple, Orange, Banana*. By not involving negation in quantifier scoping, a semantic parser may produce an unintended interpretation.

Previous work on statistical QSD has been quite restricted. Higgins and Sadock (2003), which we refer to as **HS03**, developed the first statistical QSD system for English. Their system disambiguates the scope of exactly two explicitly quantified NPs in a sentence, ignoring indefinite *a/an*, definites and bare NPs. Manshadi and Allen (2011a), hence **MA11**, go beyond those limitations and scope an arbitrary number of NPs in a sentence with no restriction on the type of quantification. However, although their corpus annotates the scope of negations and the implicit universal of plurals, their QSD system does not handle those.

As a step towards comprehensive automatic QSD, in this paper we present our work on automatic scoping of the implicit universal of plurals and negations. For data, we use a new revision of MA11’s corpus, first introduced in Manshadi et al. (2011b). The new revision, called **QuanText**, carries a more detailed, fine-grained scope annotation (Manshadi et al., 2012). The performance of

<sup>2</sup>Although plurals carry different types of quantification (Herbelot and Copestake, 2010), almost always there exists an implicit universal. The importance of scoping this universal, however, may vary based on the type of quantification.

our model defines a baseline for future efforts on (comprehensive) QSD over QuanText. In addition to addressing plurality and negation, this work improves upon MA11’s in two directions.

- We theoretically justify MA11’s ternary-classification approach, formulating it as a general framework for learning to build partial orders. An  $n \log n$  algorithm is then given to find a guaranteed approximation within a fixed ratio of the optimal solution from a set of pairwise preferences (Sect. 3.1).
- We replace MA11’s hand-annotated features with a set of automatically generated linguistic features. Our rich set of features significantly improves the performance of the QSD model, even though we give up the gold-standard dependency features (Sect. 3.3).

## 2 Task definition

In QuanText, scope-bearing elements (or, as we call them, scopal terms) of each sentence have been identified using labeled **chunks**, as in (3).

3. *Replace [1/ every line] in [2/ the file] ending in [3/ punctuation] with [4/ a blank line].*

NP chunks follow the definition of **baseNP** (Ramshaw and Marcus, 1995) and hence are flat. Outscoping relations are used to specify the relative scope of scopal terms. The relation  $i > j$  means that chunk  $i$  **outscopes** (or has **wide scope** over) chunk  $j$ . Equivalently, chunk  $j$  is said to have **narrow scope** with respect to  $i$ . Each sentence is annotated with its most preferred scoping (according to the annotators’ judgement), represented as a **partial order**:

4.  $SI : (2 > 1 > 4; 1 > 3)$

If neither  $i > j$  nor  $j > i$  is entailed from the scoping,  $i$  and  $j$  are **incomparable**. This happens if both orders are equivalent (as in two existentials) or when the two chunks have no scope interaction.

Since a partial order can be represented by a **Directed Acyclic Graph** (DAG), we use DAGs to represent scopings. For example,  $G_1$  in Figure 1 represents the scoping in (4).

### 2.1 Evaluation metrics

Given the gold standard DAG  $G_g = (V, E_g)$  and the predicted DAG  $G_p = (V, E_p)$ , a similarity measure may be defined based on the ratio of the number of pairs (of nodes) labeled correctly to the

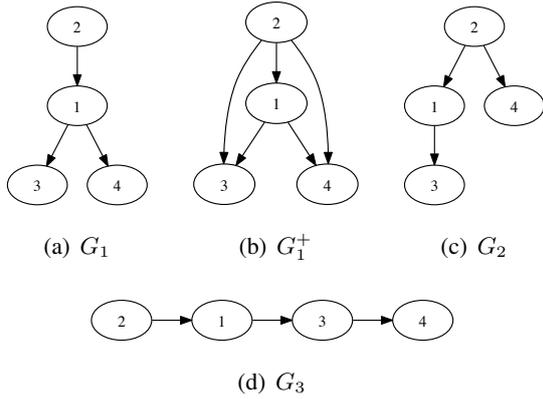


Figure 1: Scoping as DAG

total number of pairs. In order to take the transitivity of outscoping relations into account, we use the **transitive closure** (TC) of DAGs. Let  $G^+ = (V, E^+)$  represent the TC of a DAG  $G = (V, E)$ .<sup>3</sup>  $G_1$  and  $G_1^+$  in Figure 1 illustrate this concept. We now define the similarity metric  $S^+$  as follows:

$$\sigma^+ = \frac{|E_p^+ \cap E_g^+| \cup |E_p^{\bar{+}} \cap E_g^{\bar{+}}|}{|V|(|V| - 1)/2} \quad (1)$$

in which  $\bar{G} = (V, \bar{E})$  is the complement of the underlying undirected version of  $G$ .

HS03 and others have used such a similarity measure for evaluation purposes. A disadvantage of this metric is that it gives the same weight to outscoping and incomparability relations. In practice, if two scopal terms with equivalent ordering (and hence, no outscoping relation) are incorrectly labeled with an outscoping, the logical form still remains valid. But if an outscoping relation is mislabeled, it will change the interpretation of the sentence. Therefore, in MA11, we suggest defining a precision/recall based on the number of outscoping relations recovered correctly:<sup>4</sup>

$$P^+ = \frac{|E_p^+ \cap E_g^+|}{|E_p^+|}, R^+ = \frac{|E_p^+ \cap E_g^+|}{|E_g^+|} \quad (2)$$

<sup>3</sup>  $(u, v) \in G^+ \iff ((u, v) \in G \vee \exists w_1 \dots w_n \in V, (u, w_1) \dots (w_n, v) \in E)$

<sup>4</sup>MA11 argues that TC-based metrics tend to produce higher numbers. For example if  $G_3$  in Figure 1 is a gold-standard DAG and  $G_1$  is a candidate DAG, TC-based metrics count  $2 > 3$  as another match, even though it is entailed from  $2 > 1$  and  $1 > 3$ . They give an alternative metric based on **transitive reduction** (TR), obtained by removing all the redundant edges of a DAG. TR-based metrics, however, have their own disadvantage. For example, if  $G_2$  is another candidate for  $G_3$ , TR-based metrics produce the same numbers for both  $G_1$  and  $G_2$ , even though  $G_1$  is clearly closer to  $G_3$  than  $G_2$ . Therefore, in this paper we stick to TC-based metrics.

### 3 Our framework

#### 3.1 Learning to do QSD

Since we defined QSD as a partial ordering, automatic QSD would become the problem of learning to build partial orders. The machine learning community has studied the problem of learning **total orders (ranking)** in depth (Cohen et al., 1999; Furnkranz and Hullermeier, 2003; Hullermeier et al., 2008). Many ranking systems create partial orders as output when the confidence level for the relative order of two objects is below some threshold. However, the target being a partial order is a fundamentally different problem. While the lack of order between two elements is interpreted as the lack of confidence in the former, it should be interpreted as **incomparability** in the latter. Learning to build partial orders has not attracted much attention in the learning community, although as seen shortly, the techniques developed for ranking can be adopted for learning to build partial orders.

As mentioned before, a partial order  $P$  can be represented by a DAG  $G$ , with  $a$  preceding  $b$  in  $P$  if and only if  $a$  reaches  $b$  in  $G$  by a directed path. Although there could be many DAGs representing a partial order  $P$ , only one of those is a **transitive DAG**.<sup>5</sup> Therefore, in order to have a one-to-one relationship between QSDs and DAGs, we only consider the class of transitive DAGs, or **TDAG**. Every non-transitive DAG will be converted into its transitive counterpart by taking its transitive closure (as shown in Figure 1).

Consider  $V$ , a set of nodes and a TDAG  $G = (V, E)$ . It would help to think of disconnected nodes  $u, v$  of  $G$ , as connected with a null edge  $\epsilon$ . We define the labeling function  $\delta_G : V \times V \rightarrow \{+, -, \epsilon\}$  assigning one of the three labels to each pair of nodes in  $G$ :

$$\delta_G(u, v) = \begin{cases} + & (u, v) \in G \\ - & (v, u) \in G \\ \epsilon & \text{otherwise} \end{cases} \quad (3)$$

Given the true TDAG  $\hat{G} = (V, \hat{E})$ , and a candidate TDAG  $G$ , we define the **Loss** function to be the total number of incorrect edges:

$$L(G, \hat{G}) = \sum_{u \prec v \in V} I(\delta_G(u, v) \neq \delta_{\hat{G}}(u, v)) \quad (4)$$

in which  $\prec$  is an arbitrary total order over the nodes in  $V$ ,<sup>6</sup> and  $I(\cdot)$  is the indicator function. We

<sup>5</sup> $G$  is transitive iff  $(u, v), (v, w) \in G \implies (u, w) \in G$ .

<sup>6</sup>E.g., the left-to-right order of the corresponding chunks in the sentence.

adopt a **minimum Bayes risk** (MBR) approach, with the goal of finding the graph with the lowest expected loss against the (unknown) target graph:

$$G^* = \operatorname{argmin}_{G \in \text{TDAG}} E_{\hat{G}} [L(G, \hat{G})] \quad (5)$$

Substituting in the definition of the loss function and exchanging the order of the expectation and summation, we get:

$$\begin{aligned} G^* &= \operatorname{argmin}_{G \in \text{TDAG}} \sum_{u \prec v \in V} E_{\hat{G}} [I(\delta_G(u, v) \neq \delta_{\hat{G}}(u, v))] \\ &= \operatorname{argmin}_{G \in \text{TDAG}} \sum_{u \prec v \in V} P(\delta_G(u, v) \neq \delta_{\hat{G}}(u, v)) \quad (6) \end{aligned}$$

This means that in order to solve Eq. (5), we need only the probabilities of each of the three labels for each of the  $C(n, 2) = n(n-1)/2$  pairs of nodes<sup>7</sup> in the graph, rather than a probability for each of the superexponentially many possible graphs. We train a classifier to estimate these probabilities directly for a given pair. Therefore, we have reduced the problem of predicting a partial order to **pairwise comparison**, analogous to ranking by pairwise comparison or **RPC** (Hullermeier et al., 2008; Furnkranz and Hullermeier, 2003), a popular technique in learning total orders. The difference though is that in RPC, the comparison is a (soft) binary classification, while for partial orders we have the case of incomparability (the label  $\epsilon$ ), hence a (soft) ternary classification.

A soft ternary classifier generates three probabilities,  $p_{u,v}(+)$ ,  $p_{u,v}(-)$ , and  $p_{u,v}(\epsilon)$  for each pair  $(u, v)$ ,<sup>8</sup> corresponding to the three labels. Hence, equation Eq. (6) can be rearranged as follows:

$$G^* = \operatorname{argmax}_{G \in \text{TDAG}} \sum_{u \prec v \in V} p_{u,v}(\delta_G(u, v)) \quad (7)$$

Let  $\Gamma_p$  be a graph like the one in Figure 2, containing exactly three edges between every two nodes, weighted by the probabilities from the  $n(n-1)/2$  classifiers. We call  $\Gamma_p$  the **preference graph**. Intuitively speaking, the solution to Eq. (7) is the transitive directed acyclic subgraph of  $\Gamma_p$  that has the maximum sum of weights. Unfortunately finding this subgraph is an NP-hard problem.<sup>9</sup>

<sup>7</sup>Throughout this subsection, unless otherwise specified, by a pair of nodes we mean a pair  $(u, v)$  with  $u \prec v$ .

<sup>8</sup> $p_{v,u}$  for  $u \prec v$  is defined in the obvious way:  $p_{v,u}(+) = p_{u,v}(-)$ ,  $p_{v,u}(-) = p_{u,v}(+)$ , and  $p_{v,u}(\epsilon) = p_{u,v}(\epsilon)$ .

<sup>9</sup>The proof is beyond the scope of this paper, but the idea is similar to that of Cohen et al. (1999), on finding total orders. Although they don't use an RPC technique, Cohen et

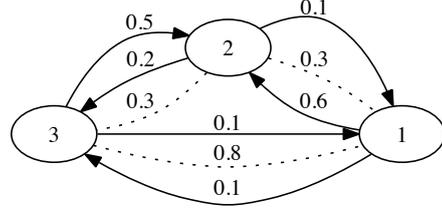


Figure 2: A preference graph over three nodes.

1. Let  $\Gamma_p$  be the preference graph and set  $G$  to  $\emptyset$ .
2.  $\forall u \in V$ , let  $\pi(u) = \sum_v p_{u,v}(+) - \sum_v p_{u,v}(-)$ .
3. Let  $u^* = \operatorname{argmax}_u \pi(u)$ ,  $S^- = \sum_{v \in G} p_{v,u^*}(-)$  &  $S^\epsilon = \sum_{v \in G} p_{v,u^*}(\epsilon)$ .
4. Remove  $u^*$  and all its incident edges from  $\Gamma_p$ .
5. Add  $u^*$  to  $G$ ; also if  $S^- > S^\epsilon$ , for every  $v \in G - u^*$ , add  $(v, u^*)$  to  $G$ .
6. If  $\Gamma_p$  is empty, output  $G$ , otherwise repeat steps 2-5.

Figure 3: An approximation algorithm for Eq. (7)

Since it is very unlikely to find an efficient algorithm to solve Eq. (7), instead, we propose the algorithm in Figure 3 which finds an approximate solution. The idea of the algorithm is simple. By finding  $u^*$  with the highest  $\pi(u)$  in step 3, we form a **topological order** for the nodes in  $G$  in a greedy way (see Footnote 9). We then add  $u^*$  to  $G$ . A directed edge is added either from every node in  $G - u^*$  to  $u^*$  or from no node, depending on which case makes the sum of the weights in  $G$  higher.

**Theorem 1** *The algorithm in Figure 3 is a 1/3-OPT approximation algorithm for Eq. (7).*

*Proof idea.* First of all, note that  $G$  is a TDAG, because edges are only added to the most recently created node in step 5. Let  $OPT$  be the optimum value of the right hand side of Eq. (7). The sum of all the weights in  $\Gamma_p$  is an upper bound for  $OPT$ :

$$\sum_{u \prec v \in V} \sum_{\lambda \in \{+, -, \epsilon\}} p_{u,v}(\lambda) \geq OPT$$

Step 5 of the algorithm guarantees that the labels  $\delta_G(u, v)$  satisfy:

$$\sum_{u \prec v \in V} p_{u,v}(\delta_G(u, v)) \geq \sum_{u \prec v \in V} p_{u,v}(\lambda) \quad (8)$$

al. (1999) encounter a similar optimization problem. They propose an approximation algorithm which finds the solution (a total order) in a greedy way. Here we use the same greedy technique to find a total order, but take it only as the topological order of the solution (Figure 3).

for any  $\lambda \in \{+, -, \epsilon\}$ . Hence:

$$\begin{aligned} \sum_{u \prec v \in V} p_{u,v}(\delta_G(u, v)) &= \frac{1}{3} \left( 3 \sum_{u \prec v \in V} p_{u,v}(\delta_G(u, v)) \right) \\ &\geq \frac{1}{3} \sum_{u \prec v \in V} \sum_{\lambda \in \{+, -, \epsilon\}} p_{u,v}(\lambda) \\ &\geq \frac{1}{3} OPT \end{aligned}$$

In practice, we improve the algorithm in Figure 3, while maintaining the approximation guarantee, as follows. When adding a node  $u^*$  to graph  $G$ , we do not make a binary decision as to whether connect every node in  $G$  to  $u^*$  or none, but we use some heuristics to choose a subset of nodes (possibly empty) in  $G$  that if connected to  $u^*$  results in a TDAG whose sum of weights is at least as big as the binary none-vs-all case. As described in Sec. 4, the algorithm works very well in our QSD system, finding the optimum solution in virtually all cases we examined.

### 3.2 Dealing with plurality and negation

Consider the following sentence with the plural NP chunk *the lines*.

5. Merge [**1p**/ the lines], ending in [**2**/ a punctuation], with [**3**/ the next non-blank line].

6. *SI* : ( $1c > 1d > 2$ ;  $1d > 3$ )<sup>10</sup>

In QuanText, plural chunks are indexed with a number followed by the lowercase letter “p”. As seen in (6), the scoping looks different from before in that the terms *1d* and *1c* are not the label of any chunk. These two terms refer to the two quantified terms introduced by the plural chunk *1p*: *1c* (for **collection**) represents the set (or in better words collection) of entities, defined by the plural, and *1d* (for **distribution**) refers to the implicit universal, introduced by the plural. In other words, for a plural chunk *ip*, *id* represents the universally quantified entity over the collection *ic*. The outscoping relation  $1d > 2$  in (6) states that every *line* in the collection, denoted by *1c*, starts with its own *punctuation* character. Similarly,  $1d > 3$  indicates that every *line* has its own *next non-blank line*. Figure 4(a) shows a DAG for the scoping in (6).

In (7) we have a sentence containing a negation. In QuanText, negation chunks are labeled with an uppercase “N” followed by a number.

<sup>10</sup>This scoping corresponds to the logical formula:  
 $\mathcal{D}x_{1c}, \text{Collection}(x_{1c}) \wedge \forall x_{1d}, \text{In}(x_{1d}, x_{1c}) \Rightarrow$   
 $(\text{Line}(x_{1d}) \wedge (\exists x_2, \text{Punctuation}(x_2) \wedge \text{EndIn}(x_{1d}, x_2)) \wedge$   
 $(\mathcal{D}x_3, \neg \text{blank}(x_3) \wedge \text{next}(x_{1d}, x_3) \wedge \text{merge}(x_{1d}, x_3)))$   
It is straightforward to write a formula for, say,  $1c > 2 > 1d$ .

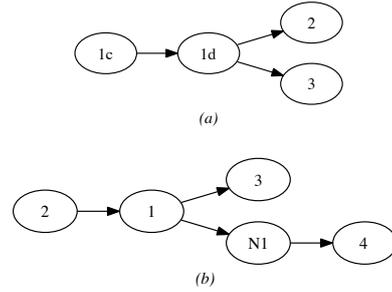


Figure 4: DAGs for scopings in (6) and (8)

7. Extract [**1**/ every word] in [**2**/ file “1.txt”], which starts with [**3**/ a capital letter], but does [**N1**/ not] end with [**4**/ a capital letter].
8. *SI* : ( $2 > 1 > 3$ ;  $1 > N1 > 4$ )

As seen here, a negation simply introduces a chunk, which participates in outscoping relations like an NP chunk. Figure 4(b) represents the scoping in (8) as a DAG.

From these examples, as long as we create two nodes in the DAG corresponding to each plural chunk, and one node corresponding to each negation, there is no need to modify the underlying model (defined in the previous section). However, when  $u$  (or  $v$ ) is a negation (*Ni*) or an implicit universal (*id*) node, the probabilities  $p_{u,v}^\lambda$  ( $\lambda \in \{+, -, \epsilon\}$ ) may come from a different source, e.g. a different classification model or the same model with a different set of features, as described in the following section.

### 3.3 Feature selection

Previous work has shown that the lexical item of quantifiers and syntactic clues (often extracted from phrase structure trees) are good at predicting quantifier scoping. Srinivasan and Yates (2009) use the semantics of the head noun in a quantified NP to predict the scoping. MA11 also find the lexical item of the head noun to be a good predictor. In this paper, we introduce a new set of syntactic features which we found very informative: the “type” dependency features of de Marneffe et al. (2006). Adopting this new set of features, we outperform MA11’s system by a large margin. Another point to mention here is that the features that are predictive of the relative scope of quantifiers are not necessarily as helpful when determining the scope of negation and vice versa. Therefore we do not use exactly the same set of features when

one of the scopal terms in the pair<sup>11</sup> is a negation, although most of the features are quite similar.

### 3.3.1 NP chunks

We first describe the set of features we have adopted when both scopal terms in a pair are NP-chunks. We have organized the features into different categories listed below.

#### Individual NP-chunk features

Following features are extracted for both NP chunks in a pair.

- The part-of-speech (POS) tag of the head of chunk
- The lexical item of the head noun
- The lexical item of the determiner/quantifier
- The lexical item of the pre-determiner
- Does the chunk contain a constant (e.g. “do”, ‘x’)?
- Is the NP-chunk a plural?

#### Implicit universal of a plural

Remember that every plural chunk  $i$  introduces two nodes in the DAG,  $ic$  and  $id$ . Both nodes are introduced by the same chunk  $i$ , therefore they use the same set of features. The only exception is a single additional binary feature for plural NP chunks, which determines whether the given node refers to the implicit universal of the plural (i.e.  $id$ ) or to the collection itself (i.e.  $ic$ ).

- Does this node refer to an implicit universal?

#### Syntactic features – phrase structure tree

As mentioned above, we have used two sets of syntactic features. The first is motivated by HS03’s work and is based on the constituency (i.e. phrase structure) tree  $T$  of the sentence. Since our model is based on pairwise comparison, the following features are defined for each pair of chunks. In the following, by chunk we mean the deepest phrase-level node in  $T$  dominating all the words in the chunk. If the constituency tree is correct, this node is usually an NP node. Also,  $P$  refers to the undirected path in  $T$  connecting the two chunks.

- Syntactic category of the deepest common ancestor
- Does 1st/2nd chunk  $C$ -command 2nd/1st one?
- Length of the path  $P$
- Syntactic categories of nodes on  $P$
- Is there a conjoined node on  $P$ ?
- List of punctuation marks dominated by nodes on  $P$

#### Syntactic features – dependency tree

Although regular “untyped” dependency relations do not seem to help our QSD system in the presence of phrase-structure trees, we found the *col-*

<sup>11</sup>Since our model is based on pairwise comparison, every sample is in fact a pair of nodes  $(u, v)$  of the DAG.

*lapsed typed dependencies* (de Marneffe and Manning, 2008) very helpful, even when used on top of the phrase-structure features. Below is the list of features we extract from the collapsed typed dependency tree  $T_d$  of each sentence. In the following, by noun we mean the node in  $T_d$  which corresponds to the head of the chunk. The choice of the word *noun*, however, may be sloppy, as the head of an NP chunk may not be a noun.

- Does 1st/2nd noun dominate 2nd/1st noun?
- Does 1st/2nd noun immediately dominate 2nd/1st?
- Type of incoming dependency relation of each noun
- Syntactic category of the deepest common ancestor
- Lexical item of the deepest common ancestor
- Length of the undirected path between the two

### 3.3.2 Negations

There are no sentences in our corpus with more than one negation. Therefore, for every pair of nodes with one negation, the other node must refer to an NP chunk. We use the following word-level, phrase-structure, and dependency features for these pairs.

- Lexical item of the determiner for the NP chunk
- Does the NP chunk contain a constant?
- Is the NP chunk a plural?
- If so, does this node refer to its implicit universal?
- Does the negation  $C$ -command the NP chunk in  $T$ ?
- Does the NP chunk  $C$ -command the negation in  $T$ ?
- What is the POS of the parent  $p$  of negation in  $T_d$ ?
- Does  $p$  dominate the noun in  $T_d$ ?
- Does the noun dominate  $p$  in  $T_d$ ?
- Does  $p$  immediately dominate the noun in  $T_d$ ?
- If so, what is the type of the dependency?
- Does the noun immediately dominate  $p$  in  $T_d$ ?
- If so, what is the type of the dependency?
- Length of the undirected path between the two in  $T_d$

## 4 Experiments

QuantText contains 500 sentences with a total of 1750 chunks, that is 3.5 chunks/sentence on average. Of those, 1700 chunks are NP chunks. The rest are scopal operators, mainly negation. Of all the NP chunks, 320 (more than 18%) are plural, each introducing an implicit universal, that is, an additional node in the DAG. Since we feed each pair of elements to the classifiers independently, each (unordered) pair introduces one sample. Therefore, a sentence with  $n$  scopal elements creates  $C(n, 2) = n(n - 1)/2$  samples for classification. When all the elements are taken into account,<sup>12</sup> the total number of samples in the corpus will be:

<sup>12</sup>Here by all elements we mean explicit chunks and the implicit universals. QuantText labels some other (implicit) elements, which we have not been handled in this work. In particular, some nouns introduce two entities: a **type** and a

$$\sum_i C(n_i, 2) \approx 4500 \quad (9)$$

Where  $n_i$  is the number of scopal terms introduced by sentence  $i$ . Out of the 4500 samples, around 1800 involve at least one implicit universal (i.e., *id*), but only 120 samples contain a negation. We evaluate the performance of the system for implicit universals and negation both separately and in the context of full scope disambiguation. We split the corpus at random into three sets of 50, 100, and 350 sentences, as development, test, and train sets respectively.<sup>13</sup>

To extract part-of-speech tags, phrase structure trees, and typed dependencies, we use the Stanford parser (Klein and Manning, 2003; de Marneffe et al., 2006) on both train and test sets. Since we are using SVM, we have passed the confidence levels through a softmax function to convert them into probabilities  $P_{u,v}^\lambda$  before applying the algorithm of Section 3. We take MA11’s system as the baseline. However, in order to have a fair comparison, we have used the output of the Stanford parser to automatically generate the same features that MA11 have hand-annotated.<sup>14</sup> In order to run the baseline system on implicit universals, we take the feature vector of a plural NP and add a feature to indicate that this feature vector represents the implicit universal of the corresponding chunk. Similarly, for negation we add a feature to show that the chunk represents a negation. As shown in Section 3.3.2, we have used a more compact set of features for negations. Once again, in order to have a fair comparison, we apply a similar modification to the baseline system. We also use the exact same classifier as used in MA11.<sup>15</sup> Figure 5(a) compares the performance of our model, which we refer to as RPC-SVM-13, with the baseline system, but only on explicit NP chunks.<sup>16</sup> The goal for running this experiment has been to compare the performance of our model to the baseline system, as described by Manshadi et al. (2012). In this work, we have only considered the token entity introduced by those nouns and have ignored the type entity.

<sup>13</sup>Since the percentage of sentences with negation is small, we made sure that those sentences are distributed uniformly between three sets.

<sup>14</sup>MA11’s features are similar to part-of-speech tags and typed dependency relations.

<sup>15</sup>*SVM<sup>Multiclass</sup>* from *SVM-light* (Joachims, 1999).

<sup>16</sup>In all experiments, we ignore NP conjunctions. Previous work treats a conjunction of NPs as separate NPs. However, similar to plurals, NP conjunctions (disjunctions) introduce an extra scopal element: a universal (existential). We are working on an annotation scheme for NP conjunctions, so we have left this for after the annotations become available.

NP-Chunks only (no <i>id</i> or negation)	$\sigma^+$	$P^+$	$R^+$	$F^+$	AR	A
Baseline (MA11)	0.762	0.638	0.484	0.550	0.59	0.47
Our model (RPC-SVM-13)	0.827	0.743	0.677	0.709	0.68	0.55

(a) Scoping explicit NP chunks

Overall system (including negation and implicit universals)	$\sigma^+$	$P^+$	$R^+$	$F^+$	AR	A
Baseline (MA11)	0.787	0.688	0.469	0.557	0.59	0.47
Our model (RPC-SVM-13)	0.863	0.784	0.720	0.751	0.69	0.55

(b) Scoping all elements (including *id* and  $N_i$ )

Figure 5: Performance on QuanText data

tem on the task that it was actually defined to perform (that is scoping only explicit NP chunks).

As seen in this table, by incorporating a richer set of features and a better learning algorithm, our model outperforms the baseline by almost 15%. The measure  $A$  in these figures shows sentence-based accuracy. A sentence counts as correct iff every pair of scopal elements has been labeled correctly. Therefore  $A$  is a tough measure. Furthermore, it is sensitive to the length of the sentence. Following MA11, we have computed another sentence-based accuracy measure,  $AR$ . In computing  $AR$ , a sentence counts as correct iff all the outscoping relations have been recovered correctly – in other words, iff  $R = 100\%$ , regardless of the value of  $P$ .  $AR$  may be more practically meaningful, because if in the correct scoping of the sentence there is no outscoping between two elements, inserting one does not affect the interpretation of the sentence. In other words, precision is less important for QSD in practice.

Figure 5(b) gives the performance of the overall model when all the elements including the implicit universals and the negations are taken into account. That the F-score of our model for the second experiment is 0.042 higher than F-score for the first indicates that scoping implicit universals and/or negations must be easier than scoping explicit NP chunks. In order to find how much one or both of the two elements contribute to this gain, we have run two more experiments, scoping only the pairs with at least one implicit universal and pairs with one negation, respectively. Figure 6 reports the results. As seen, the contribution in boosting the overall performance comes from the implicit universals while negations, in fact, lower the performance. The performance for pairs with implicit universal is higher because universals, in general,

<b>Implicit universals only</b> (pairs with at least one <i>id</i> )	<b>P<sup>+</sup></b>	<b>R<sup>+</sup></b>	<b>F<sup>+</sup></b>
Baseline (MA11)	0.776	0.458	0.576
Our model (RPC-SVM-13)	0.836	0.734	0.782

(a) Pairs with at least one implicit universal

<b>Negation only</b> (pairs with one negation)	<b>P<sup>+</sup></b>	<b>R<sup>+</sup></b>	<b>F<sup>+</sup></b>
Baseline (MA11)	0.502	0.571	0.534
Our model (RPC-SVM-13)	0.733	0.55	0.629

(b) Pairs with at least one negation

Figure 6: Implicit universals and negations

are easier to scope, even for the human annotators.<sup>17</sup> There are several reasons for poor performance with negations as well. First, the number of negations in the corpus is small, therefore the data is very sparse. Second, the RPC model does not work well for negations. Scoping a negation relative to an NP chunk, with which it has a long distance dependency, often depends on the scope of the elements in between. Third, scoping negation usually requires a deep semantic analysis.

In order to see how well our approximation algorithm is working, similar to the approach of Chambers and Jurafsky (2008), we tried an ILP solver<sup>18</sup> for DAGs with at most 8 nodes to find the optimum solution, but we found the difference insignificant. In fact, the approximation algorithm finds the optimum solution in all but one case.<sup>19</sup>

## 5 Related work

Since automatic QSD is in general challenging, traditionally quantifier scoping is left underspecified in deep linguistic processing systems (Alshawi and Crouch, 1992; Bos, 1996; Copestake et al., 2001). Some efforts have been made to move underspecification frameworks towards weighted constraint-based graphs in order to produce the most preferred reading (Koller et al., 2008), but the source of these types of constraint are often discourse, pragmatics, world knowledge, etc., and hence, they are hard to obtain automatically. In or-

<sup>17</sup>Trivially, we have taken the relation outscoping  $ic > id$  for granted and not counted it towards higher performance.

<sup>18</sup>lpsolve: <http://sourceforge.net/projects/lpsolve>

<sup>19</sup>To find the gain that can be obtained with gold-standard parses, we used MA11’s system with their hand-annotated and the equivalent automatically generated features. The former boost the performance by 0.04. Incidentally, HS03 lose almost 0.04 when switching to automatically generated parses.

der to evade scope disambiguation, yet be able to perform entailment, Koller and Thater (2010) propose an algorithm to calculate the weakest readings<sup>20</sup> from a scope-underspecified representation.

Early efforts on automatic QSD (Moran, 1988; Hurum, 1988) were based on heuristics, manually formed into rules with manually assigned weights for resolving conflicts. To the best of our knowledge, there have been four major efforts on statistical QSD for English: Higgins and Sadock (2003), Galen and MacCartney (2004), Srinivasan and Yates (2009), and Manshadi and Allen (2011a). The first three only scope two scopal terms in a sentence, where the scopal term is an NP with an explicit quantification. MA11 is the first to scope any number of NPs in a sentence with no restriction on the type of quantification. Besides ignoring negation and implicit universals, their system has some other limitations too. First, the learning model is not theoretically justified. Second, hand-annotated features (e.g. dependency relations) are used on both the train and the test data.

## 6 Summary and future work

We develop the first statistical QSD model addressing the interaction of quantifiers with negation and the implicit universal of plurals, defining a baseline for this task on QuanText data (Manshadi et al., 2012). In addition, our work improves upon Manshadi and Allen (2011a)’s work by (approximately) optimizing a well justified criterion, by using automatically generated features instead of hand-annotated dependencies, and by boosting the performance by a large margin with the help of a rich feature vector.

This work can be improved in many directions, among which are scoping more elements such as other scopal operators and implicit entities, deploying more complex learning models, and developing models which require less supervision.

## Acknowledgement

We need to thank William de Beaumont and Jonathan Gordon for their comments on the paper and Omid Bakhshandeh for his assistance. This work was supported in part by NSF grant 1012205, and ONR grant N000141110417.

<sup>20</sup>Those which can be entailed from other readings but do not entail any other reading

## References

- Hiyan Alshawi and Richard Crouch. 1992. Monotonic semantic interpretation. In *Proceedings of Association for Computational Linguistics*, pages 32–39.
- Johan Bos. 1996. Predicate logic unplugged. In *Proceedings of the 10th Amsterdam Colloquium*, pages 133–143.
- Nathanael Chambers and Dan Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 698–706, Stroudsburg, PA.
- William W. Cohen, Robert E. Schapire, and Yoram Singer. 1999. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270.
- Ann Copestake, Alex Lascarides, and Dan Flickinger. 2001. An algebra for semantic construction in constraint-based grammars. In *Proceedings of Association for Computational Linguistics '01*, pages 140–147.
- Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation, CrossParser '08*, pages 1–8.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure trees. In *Proceedings of International Conference on Language Resources and Evaluation '12*.
- Johannes Furnkranz and Eyke Hullermeier. 2003. Pairwise preference learning and ranking. In *Proceedings of the 14th European Conference on Machine Learning*, volume 2837, pages 145–156.
- Andrew Galen and Bill MacCartney. 2004. Statistical resolution of scope ambiguity in natural language. <http://nlp.stanford.edu/nlkr/scoper.pdf>.
- Fritz Hamm and Edward W. Hinrichs. 2010. *Plurality and Quantification*. Studies in Linguistics and Philosophy. Springer.
- Aurelie Herbelot and Ann Copestake. 2010. Annotating underquantification. In *Proceedings of the Fourth Linguistic Annotation Workshop, LAW IV '10*, pages 73–81.
- Derrick Higgins and Jerrold M. Sadock. 2003. A machine learning approach to modeling scope preferences. *Computational Linguistics*, 29(1):73–96.
- Eyke Hullermeier, Johannes Furnkranz, Weiwei Cheng, and Klaus Brinker. 2008. Label ranking by learning pairwise preferences. *Artificial Intelligence*, 172(1617):1897 – 1916.
- Sven Hurum. 1988. Handling scope ambiguities in English. In *Proceedings of the second conference on Applied natural language processing, ANLC '88*, pages 58–65.
- Thorsten Joachims. 1999. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in kernel methods*, pages 169–184. MIT Press, Cambridge, MA, USA.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03*, pages 423–430.
- Alexander Koller and Stefan Thater. 2010. Computing weakest readings. In *Proceedings of the 48th Annual Meeting on Association for Computational Linguistics*, Uppsala, Sweden.
- Alexander Koller, Michaela Regneri, and Stefan Thater. 2008. Regular tree grammars as a formalism for scope underspecification. In *Proceedings of Annual Meeting on Association for Computational Linguistics and Human Language Technologies '08*.
- Fred Landmann. 2000. *Events and plurality*. Kluwer Academic Publishers, Dordrecht.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of Association for Computational Linguistics (ACL)*.
- Mehdi Manshadi and James Allen. 2011a. Unrestricted quantifier scope disambiguation. In *Proceedings of Association for Computational Linguistics '11, Workshop on Graph-based Methods for NLP (TextGraph-6)*.
- Mehdi Manshadi, James Allen, and Mary Swift. 2011b. A corpus of scope-disambiguated English text. In *Proceedings of Association for Computational Linguistics and Human Language Technologies '11: short papers*, pages 141–146.
- Mehdi Manshadi, James Allen, and Mary Swift. 2012. An annotation scheme for quantifier scope disambiguation. In *Proceedings of International Conference on Language Resources and Evaluation '12*.
- Douglas Moran. 1988. Quantifier scoping in the SRI core language engine. In *Proceedings of the 26th Annual Meeting on Association for Computational Linguistics*.
- Lance Ramshaw and Mitch Marcus. 1995. Text Chunking Using Transformation-Based Learning. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey.
- Prakash Srinivasan and Alexander Yates. 2009. Quantifier scope disambiguation using extracted pragmatic knowledge: preliminary results. In *Proceedings of EMNLP '09*.

# Joint Event Extraction via Structured Prediction with Global Features

Qi Li    Heng Ji    Liang Huang

Departments of Computer Science and Linguistics

The Graduate Center and Queens College

City University of New York

New York, NY 10016, USA

{liqiearth, hengjicuny, liang.huang.sh}@gmail.com

## Abstract

Traditional approaches to the task of ACE event extraction usually rely on sequential pipelines with multiple stages, which suffer from error propagation since event triggers and arguments are predicted in isolation by independent local classifiers. By contrast, we propose a joint framework based on structured prediction which extracts triggers and arguments together so that the local predictions can be mutually improved. In addition, we propose to incorporate global features which explicitly capture the dependencies of multiple triggers and arguments. Experimental results show that our joint approach with local features outperforms the pipelined baseline, and adding global features further improves the performance significantly. Our approach advances state-of-the-art sentence-level event extraction, and even outperforms previous argument labeling methods which use external knowledge from other sentences and documents.

## 1 Introduction

Event extraction is an important and challenging task in Information Extraction (IE), which aims to discover event triggers with specific types and their arguments. Most state-of-the-art approaches (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011) use sequential pipelines as building blocks, which break down the whole task into separate subtasks, such as trigger identification/classification and argument identification/classification. As a common drawback of the staged architecture, errors in upstream component are often compounded and propagated to the downstream classifiers. The downstream components, however, cannot impact earlier deci-

sions. For example, consider the following sentences with an ambiguous word “fired”:

- (1) In Baghdad, a cameraman **died** when an *American tank* **fired** on the Palestine Hotel.
- (2) He has **fired** his *air defense chief*.

In sentence (1), “fired” is a trigger of type *Attack*. Because of the ambiguity, a local classifier may miss it or mislabel it as a trigger of *End-Position*. However, knowing that “tank” is very likely to be an *Instrument* argument of *Attack* events, the correct event subtype assignment of “fired” is obviously *Attack*. Likewise, in sentence (2), “air defense chief” is a job title, hence the argument classifier is likely to label it as an *Entity* argument for *End-Position* trigger.

In addition, the local classifiers are incapable of capturing inter-dependencies among multiple event triggers and arguments. Consider sentence (1) again. Figure 1 depicts the corresponding event triggers and arguments. The dependency between “fired” and “died” cannot be captured by the local classifiers, which may fail to attach “cameraman” to “fired” as a *Target* argument. By using global features, we can propagate the *Victim* argument of the *Die* event to the *Target* argument of the *Attack* event. As another example, knowing that an *Attack* event usually only has one *Attacker* argument, we could penalize assignments in which one trigger has more than one *Attacker*. Such global features cannot be easily exploited by a local classifier.

Therefore, we take a fresh look at this problem and formulate it, for the first time, as a *structured learning* problem. We propose a novel joint event extraction algorithm to predict the triggers and arguments simultaneously, and use the structured perceptron (Collins, 2002) to train the joint model. This way we can capture the dependencies between triggers and argument as well as explore

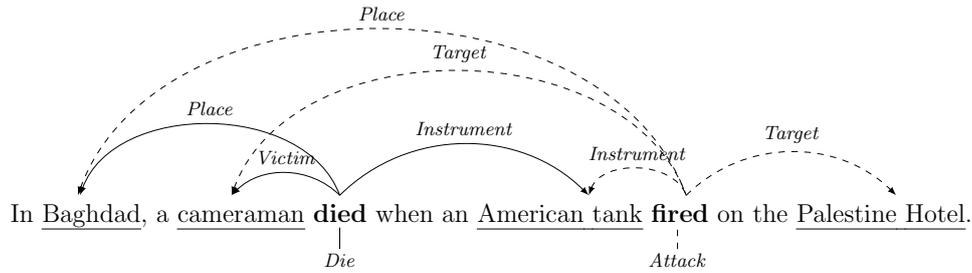


Figure 1: Event mentions of example (1). There are two event mentions that share three arguments, namely the *Die* event mention triggered by “died”, and the *Attack* event mention triggered by “fired”.

arbitrary global features over multiple local predictions. However, different from easier tasks such as part-of-speech tagging or noun phrase chunking where efficient dynamic programming decoding is feasible, here exact joint inference is intractable. Therefore we employ beam search in decoding, and train the model using the early-update perceptron variant tailored for beam search (Collins and Roark, 2004; Huang et al., 2012).

We make the following contributions:

1. Different from traditional pipeline approach, we present a novel framework for sentence-level event extraction, which predicts triggers and their arguments jointly (Section 3).
2. We develop a rich set of features for event extraction which yield promising performance even with the traditional pipeline (Section 3.4.1). In this paper we refer to them as local features.
3. We introduce various global features to exploit dependencies among multiple triggers and arguments (Section 3.4.2). Experiments show that our approach outperforms the pipelined approach with the same set of local features, and significantly advances the state-of-the-art with the addition of global features which brings a notable further improvement (Section 4).

## 2 Event Extraction Task

In this paper we focus on the event extraction task defined in Automatic Content Extraction (ACE) evaluation.<sup>1</sup> The task defines 8 event types and 33 subtypes such as *Attack*, *End-Position* etc. We introduce the terminology of the ACE event extraction that we used in this paper:

<sup>1</sup><http://projects.ldc.upenn.edu/ace/>

- Event mention: an occurrence of an event with a particular type and subtype.
- Event trigger: the word most clearly expresses the event mention.
- Event argument: an entity mention, temporal expression or value (e.g. *Job-Title*) that serves as a participant or attribute with a specific role in an event mention.
- Event mention: an instance that includes one event trigger and some arguments that appear within the same sentence.

Given an English text document, an event extraction system should predict event triggers with specific subtypes and their arguments from each sentence. Figure 1 depicts the event triggers and their arguments of sentence (1) in Section 1. The outcome of the entire sentence can be considered a graph in which each argument role is represented as a typed edge from a trigger to its argument.

In this work, we assume that argument candidates such as entities are part of the input to the event extraction, and can be from either gold standard or IE system output.

## 3 Joint Framework for Event Extraction

Based on the hypothesis that facts are interdependent, we propose to use structured perceptron with inexact search to jointly extract triggers and arguments that co-occur in the same sentence. In this section, we will describe the training and decoding algorithms for this model.

### 3.1 Structured perceptron with beam search

Structured perceptron is an extension to the standard linear perceptron for structured prediction, which was proposed in (Collins, 2002). Given a sentence instance  $x \in \mathcal{X}$ , which in our case is a sentence with argument candidates, the structured perceptron involves the following *decoding prob-*

lem which finds the best configuration  $z \in \mathcal{Y}$  according to the current model  $\mathbf{w}$ :

$$z = \underset{y' \in \mathcal{Y}(x)}{\operatorname{argmax}} \quad \mathbf{w} \cdot \mathbf{f}(x, y') \quad (1)$$

where  $\mathbf{f}(x, y')$  represents the feature vector for instance  $x$  along with configuration  $y'$ .

The perceptron learns the model  $\mathbf{w}$  in an on-line fashion. Let  $\mathcal{D} = \{(x^{(j)}, y^{(j)})\}_{j=1}^n$  be the set of training instances (with  $j$  indexing the current training instance). In each iteration, the algorithm finds the best configuration  $z$  for  $x$  under the current model (Eq. 1). If  $z$  is incorrect, the weights are updated as follows:

$$\mathbf{w} = \mathbf{w} + \mathbf{f}(x, y) - \mathbf{f}(x, z) \quad (2)$$

The key step of the training and test is the decoding procedure, which aims to search for the best configuration under the current parameters. In simpler tasks such as part-of-speech tagging and noun phrase chunking, efficient dynamic programming algorithms can be employed to perform exact inference. Unfortunately, it is intractable to perform the exact search in our framework because: (1) by jointly modeling the trigger labeling and argument labeling, the search space becomes much more complex. (2) we propose to make use of arbitrary global features, which makes it infeasible to perform exact inference efficiently.

To address this problem, we apply beam-search along with early-update strategy to perform inexact decoding. Collins and Roark (2004) proposed the early-update idea, and Huang et al. (2012) later proved its convergence and formalized a general framework which includes it as a special case. Figure 2 describes the skeleton of perceptron training algorithm with beam search. In each step of the beam search, if the prefix of oracle assignment  $y$  falls out from the beam, then the top result in the beam is returned for early update. One could also use the standard-update for inference, however, with highly inexact search the standard-update generally does not work very well because of “invalid updates”, i.e., updates that do not fix a violation (Huang et al., 2012). In Section 4.5 we will show that the standard perceptron introduces many invalid updates especially with smaller beam sizes, also observed by Huang et al. (2012).

To reduce overfitting, we used averaged parameters after training to decode test instances in our experiments. The resulting model is called averaged perceptron (Collins, 2002).

**Input:** Training set  $\mathcal{D} = \{(x^{(j)}, y^{(j)})\}_{i=1}^n$ ,  
maximum iteration number  $T$

**Output:** Model parameters  $\mathbf{w}$

```

1 Initialization: Set  $\mathbf{w} = 0$ ;
2 for  $t \leftarrow 1 \dots T$  do
3   foreach  $(x, y) \in \mathcal{D}$  do
4      $z \leftarrow \text{beamSearch}(x, y, \mathbf{w})$ 
5     if  $z \neq y$  then
6        $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{f}(x, y_{[1:|z|]}) - \mathbf{f}(x, z)$ 

```

Figure 2: Perceptron training with beam-search (Huang et al., 2012). Here  $y_{[1:i]}$  denotes the prefix of  $y$  that has length  $i$ , e.g.,  $y_{[1:3]} = (y_1, y_2, y_3)$ .

### 3.2 Label sets

Here we introduce the label sets for trigger and argument in the model. We use  $\mathcal{L} \cup \{\perp\}$  to denote the trigger label alphabet, where  $\mathcal{L}$  represents the 33 event subtypes, and  $\perp$  indicates that the token is not a trigger. Similarly,  $\mathcal{R} \cup \{\perp\}$  denotes the argument label sets, where  $\mathcal{R}$  is the set of possible argument roles, and  $\perp$  means that the argument candidate is not an argument for the current trigger. It is worth to note that the set  $\mathcal{R}$  of each particular event subtype is subject to the entity type constraints defined in the official ACE annotation guideline<sup>2</sup>. For example, the *Attacker* argument for an *Attack* event can only be one of *PER*, *ORG* and *GPE* (Geo-political Entity).

### 3.3 Decoding

Let  $x = \langle (x_1, x_2, \dots, x_s), \mathcal{E} \rangle$  denote the sentence instance, where  $x_i$  represents the  $i$ -th token in the sentence and  $\mathcal{E} = \{e_k\}_{k=1}^m$  is the set of argument candidates. We use

$$y = (t_1, a_{1,1}, \dots, a_{1,m}, \dots, t_s, a_{s,1}, \dots, a_{s,m})$$

to denote the corresponding gold standard structure, where  $t_i$  represents the trigger assignment for the token  $x_i$ , and  $a_{i,k}$  represents the argument role label for the edge between  $x_i$  and argument candidate  $e_k$ .

<sup>2</sup>[http://projects.ldc.upenn.edu/ace/docs/English-Events-Guidelines\\_v5.4.3.pdf](http://projects.ldc.upenn.edu/ace/docs/English-Events-Guidelines_v5.4.3.pdf)

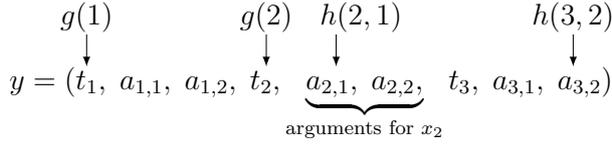


Figure 3: Example notation with  $s = 3, m = 2$ .

For simplicity, throughout this paper we use  $y_{g(i)}$  and  $y_{h(i,k)}$  to represent  $t_i$  and  $a_{i,k}$ , respectively. Figure 3 demonstrates the notation with  $s = 3$  and  $m = 2$ . The variables for the toy sentence “Jobs founded Apple” are as follows:

$$x = \langle (\underbrace{Jobs, founded, Apple}_{x_2}, \underbrace{Jobs_{PER}, Apple_{ORG}}_{\mathcal{E}}) \rangle$$

$$y = (\perp, \perp, \perp, \underbrace{Start\_Org}_{t_2}, \underbrace{Agent, Org}_{args\ for\ founded}, \perp, \perp, \perp)$$

Figure 4 describes the beam-search procedure with early-update for event extraction. During each step with token  $i$ , there are two sub-steps:

- **Trigger labeling** We enumerate all possible trigger labels for the current token. The linear model defined in Eq. (1) is used to score each partial configuration. Then the  $K$ -best partial configurations are selected to the beam, assuming the beam size is  $K$ .
- **Argument labeling** After the trigger labeling step, we traverse all configurations in the beam. Once a trigger label for  $x_i$  is found in the beam, the decoder searches through the argument candidates  $\mathcal{E}$  to label the edges between each argument candidate and the trigger. After labeling each argument candidate, we again score each partial assignment and select the  $K$ -best results to the beam.

After the second step, the rank of different trigger assignments can be changed because of the argument edges. Likewise, the decision on later argument candidates may be affected by earlier argument assignments.

The overall time complexity for decoding is  $O(K \cdot s \cdot m)$ .

### 3.4 Features

In this framework, we define two types of features, namely local features and global features. We first introduce the definition of local and global features in this paper, and then describe the implementation details later. Recall that in the linear model defined in Eq. (1),  $\mathbf{f}(x, y)$  denotes the features extracted from the input instance  $x$  along

**Input:** Instance  $x = \langle (x_1, x_2, \dots, x_s), \mathcal{E} \rangle$  and the oracle output  $y$  if for training.

$K$ : Beam size.

$\mathcal{L} \cup \{\perp\}$ : trigger label alphabet.

$\mathcal{R} \cup \{\perp\}$ : argument label alphabet.

**Output:** 1-best prediction  $z$  for  $x$

```

1 Set beam  $\mathcal{B} \leftarrow [\epsilon]$  /*empty configuration*/
2 for  $i \leftarrow 1 \dots s$  do
3    $buf \leftarrow \{z' \circ l \mid z' \in \mathcal{B}, l \in \mathcal{L} \cup \{\perp\}\}$ 
4    $\mathcal{B} \leftarrow K\text{-best}(buf)$ 
5   if  $y_{[1:g(i)]} \notin \mathcal{B}$  then
6     return  $\mathcal{B}[0]$  /*for early-update*/
7   for  $e_k \in \mathcal{E}$  do /*search for arguments*/
8      $buf \leftarrow \emptyset$ 
9     for  $z' \in \mathcal{B}$  do
10       $buf \leftarrow buf \cup \{z' \circ \perp\}$ 
11      if  $z'_{g(i)} \neq \perp$  then /* $x_i$  is a trigger*/
12         $buf \leftarrow buf \cup \{z' \circ r \mid r \in \mathcal{R}\}$ 
13       $\mathcal{B} \leftarrow K\text{-best}(buf)$ 
14      if  $y_{[1:h(i,k)]} \notin \mathcal{B}$  then
15        return  $\mathcal{B}[0]$  /*for early-update*/
16 return  $\mathcal{B}[0]$ 

```

Figure 4: Decoding algorithm for event extraction.  $z \circ l$  means appending label  $l$  to the end of  $z$ . During test, lines 4-5 & 13-14 are omitted.

with configuration  $y$ . In general, each feature instance  $f$  in  $\mathbf{f}$  is a function  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ , which maps  $x$  and  $y$  to a feature value. Local features are only related to predictions on individual trigger or argument. In the case of unigram tagging for trigger labeling, each local feature takes the form of  $f(x, i, y_{g(i)})$ , where  $i$  denotes the index of the current token, and  $y_{g(i)}$  is its trigger label. In practice, it is convenient to define the local feature function as an indicator function, for example:

$$f_1(x, i, y_{g(i)}) = \begin{cases} 1 & \text{if } y_{g(i)} = \text{Attack and } x_i = \text{“fire”} \\ 0 & \text{otherwise} \end{cases}$$

The global features, by contrast, involve longer range of the output structure. Formally, each global feature function takes the form of  $f(x, i, k, y)$ , where  $i$  and  $k$  denote the indices of the current token and argument candidate in decoding, respectively. The following indicator function is a simple example of global features:

$$f_{101}(x, i, k, y) = \begin{cases} 1 & \text{if } y_{g(i)} = \text{Attack and} \\ & \text{y has only one “Attacker”} \\ 0 & \text{otherwise} \end{cases}$$

Category	Type	Feature Description
Trigger	Lexical	<ol style="list-style-type: none"> <li>1. unigrams/bigrams of the current and context words within the window of size 2</li> <li>2. unigrams/bigrams of part-of-speech tags of the current and context words within the window of size 2</li> <li>3. lemma and synonyms of the current token</li> <li>4. base form of the current token extracted from Nomlex (Macleod et al., 1998)</li> <li>5. Brown clusters that are learned from ACE English corpus (Brown et al., 1992; Miller et al., 2004; Sun et al., 2011). We used the clusters with prefixes of length 13, 16 and 20 for each token.</li> </ol>
	Syntactic	<ol style="list-style-type: none"> <li>6. dependent and governor words of the current token</li> <li>7. dependency types associated the current token</li> <li>8. whether the current token is a modifier of job title</li> <li>9. whether the current token is a non-referential pronoun</li> </ol>
	Entity Information	<ol style="list-style-type: none"> <li>10. unigrams/bigrams normalized by entity types</li> <li>11. dependency features normalized by entity types</li> <li>12. nearest entity type and string in the sentence/clause</li> </ol>
Argument	Basic	<ol style="list-style-type: none"> <li>1. context words of the entity mention</li> <li>2. trigger word and subtype</li> <li>3. entity type, subtype and entity role if it is a geo-political entity mention</li> <li>4. entity mention head, and head of any other name mention from co-reference chain</li> <li>5. lexical distance between the argument candidate and the trigger</li> <li>6. the relative position between the argument candidate and the trigger: {before, after, overlap, or separated by punctuation}</li> <li>7. whether it is the nearest argument candidate with the same type</li> <li>8. whether it is the only mention of the same entity type in the sentence</li> </ol>
	Syntactic	<ol style="list-style-type: none"> <li>9. dependency path between the argument candidate and the trigger</li> <li>10. path from the argument candidate and the trigger in constituent parse tree</li> <li>11. length of the path between the argument candidate and the trigger in dependency graph</li> <li>12. common root node and its depth of the argument candidate and parse tree</li> <li>13. whether the argument candidate and the trigger appear in the same clause</li> </ol>

Table 1: Local features.

### 3.4.1 Local features

In general there are two kinds of local features:

**Trigger features** The local feature function for trigger labeling can be factorized as  $f(x, i, y_{g(i)}) = p(x, i) \circ q(y_{g(i)})$ , where  $p(x, i)$  is a predicate about the input, which we call text feature, and  $q(y_{g(i)})$  is a predicate on the trigger label. In practice, we define two versions of  $q(y_{g(i)})$ :

$$q_0(y_{g(i)}) = y_{g(i)} \text{ (event subtype)}$$

$$q_1(y_{g(i)}) = \text{event type of } y_{g(i)}$$

$q_1(y_{g(i)})$  is a backoff version of the standard unigram feature. Some text features for the same event type may share a certain distributional similarity regardless of the subtypes. For example, if the nearest entity mention is “*Company*”, the current token is likely to be *Personnel* no matter whether it is *End-Position* or *Start-Position*.

**Argument features** Similarly, the local feature function for argument labeling can be represented as  $f(x, i, k, y_{g(i)}, y_{h(i,k)}) = p(x, i, k) \circ q(y_{g(i)}, y_{h(i,k)})$ , where  $y_{h(i,k)}$  denotes the argument assignment for the edge between trigger word  $i$  and argument candidate  $e_k$ . We define two

versions of  $q(y_{g(i)}, y_{h(i,k)})$ :

$$q_0(y_{g(i)}, y_{h(i,k)}) = \begin{cases} y_{h(i,k)} & \text{if } y_{h(i,k)} \text{ is } Place, \\ & Time \text{ or } None \\ y_{g(i)} \circ y_{h(i,k)} & \text{otherwise} \end{cases}$$

$$q_1(y_{g(i)}, y_{h(i,k)}) = \begin{cases} 1 & \text{if } y_{h(i,k)} \neq None \\ 0 & \text{otherwise} \end{cases}$$

It is notable that *Place* and *Time* arguments are applicable and behave similarly to all event subtypes. Therefore features for these arguments are not conjuncted with trigger labels.  $q_1(y_{h(i,k)})$  can be considered as a backoff version of  $q_0(y_{h(i,k)})$ , which does not discriminate different argument roles but only focuses on argument identification. Table 1 summarizes the text features about the input for trigger and argument labeling. In our experiments, we used the Stanford parser (De Marneffe et al., 2006) to create dependency parses.

### 3.4.2 Global features

Table 2 summarizes the 8 types of global features we developed in this work. They can be roughly divided into the following two categories:

Category	Feature Description
Trigger	<ol style="list-style-type: none"> <li>1. bigram of trigger types occur in the same sentence or the same clause</li> <li>2. binary feature indicating whether synonyms in the same sentence have the same trigger label</li> <li>3. context and dependency paths between two triggers conjuncted with their types</li> </ol>
Argument	<ol style="list-style-type: none"> <li>4. context and dependency features about two argument candidates which share the same role within the same event mention</li> <li>5. features about one argument candidate which plays as arguments in two event mentions in the same sentence</li> <li>6. features about two arguments of an event mention which are overlapping</li> <li>7. the number of arguments with each role type of an event mention conjuncted with the event subtype</li> <li>8. the pairs of time arguments within an event mention conjuncted with the event subtype</li> </ol>

Table 2: Global features.

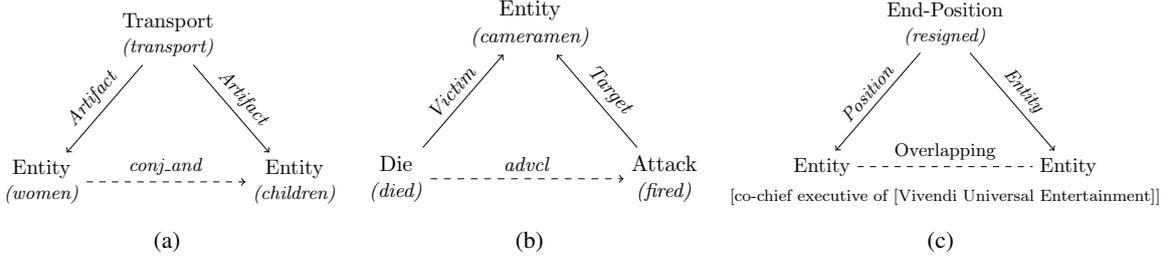


Figure 5: Illustration of global features (4-6) in Table 2.

Event	Probability
Attack	0.34
Die	0.14
Transport	0.08
Injure	0.04
Meet	0.02

Table 3: Top 5 event subtypes that co-occur with *Attack* event in the same sentence.

**Trigger global feature** This type of feature captures the dependencies between two triggers within the same sentence. For instance: feature (1) captures the co-occurrence of trigger types. This kind of feature is motivated by the fact that two event mentions in the same sentence tend to be semantically coherent. As an example, from Table 3 we can see that *Attack* event often co-occur with *Die* event in the same sentence, but rarely co-occur with *Start-Position* event. Feature (2) encourages synonyms or identical tokens to have the same label. Feature (3) exploits the lexical and syntactic relation between two triggers. A simple example is whether an *Attack* trigger and a *Die* trigger are linked by the dependency relation *conj\_and*.

**Argument global feature** This type of feature is defined over multiple arguments for the same or different triggers. Consider the following sentence:

(3) Trains running to southern Sudan were used

to *transport* abducted women and children.

The *Transport* event mention “*transport*” has two *Artifact* arguments, “*women*” and “*children*”. The dependency edge *conj\_and* between “*women*” and “*children*” indicates that they should play the same role in the event mention. The triangle structure in Figure 5(a) is an example of feature (4) for the above example. This feature encourages entities that are linked by dependency relation *conj\_and* to play the same role *Artifact* in any *Transport* event.

Similarly, Figure 5(b) depicts an example of feature (5) for sentence (1) in Section 1. In this example, an entity mention is *Victim* argument to *Die* event and *Target* argument to *Attack* event, and the two event triggers are connected by the typed dependency *advcl*. Here *advcl* means that the word “*fired*” is an adverbial clause modifier of “*died*”.

Figure 5(c) shows an example of feature (6) for the following sentence:

(4) Barry Diller *resigned* as co-chief executive of Vivendi Universal Entertainment.

The job title “*co-chief executive of Vivendi Universal Entertainment*” overlaps with the *Organization* mention “*Vivendi Universal Entertainment*”. The feature in the triangle shape can be considered as a soft constraint such that if a *Job-Title* mention is a *Position* argument to an *End-Position* trigger, then the *Organization* mention

which appears at the end of it should be labeled as *Entity* argument for the same trigger.

Feature (7-8) are based on the statistics about different arguments for the same trigger. For instance, in many cases, a trigger can only have one *Place* argument. If a partial configuration mistakenly classifies more than one entity mention as *Place* arguments for the same trigger, then it will be penalized.

## 4 Experiments

### 4.1 Data set and evaluation metric

We utilized the ACE 2005 corpus as our testbed. For comparison, we used the same test set with 40 newswire articles (672 sentences) as in (Ji and Grishman, 2008; Liao and Grishman, 2010) for the experiments, and randomly selected 30 other documents (863 sentences) from different genres as the development set. The rest 529 documents (14,840 sentences) are used for training.

Following previous work (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011), we use the following criteria to determine the correctness of an predicted event mention:

- A trigger is correct if its event subtype and offsets match those of a reference trigger.
- An argument is correctly *identified* if its event subtype and offsets match those of any of the reference argument mentions.
- An argument is correctly *identified* and *classified* if its event subtype, offsets and argument role match those of any of the reference argument mentions.

Finally we use *Precision (P)*, *Recall (R)* and *F-measure (F<sub>1</sub>)* to evaluate the overall performance.

### 4.2 Baseline system

Chen and Ng (2012) have proven that performing identification and classification in one step is better than two steps. To compare our proposed method with the previous pipelined approaches, we implemented two Maximum Entropy (Max-Ent) classifiers for trigger labeling and argument labeling respectively. To make a fair comparison, the feature sets in the baseline are identical to the local text features we developed in our framework (see Figure 1).

### 4.3 Training curves

We use the *harmonic mean* of the trigger’s  $F_1$  measure and argument’s  $F_1$  measure to measure the performance on the development set.

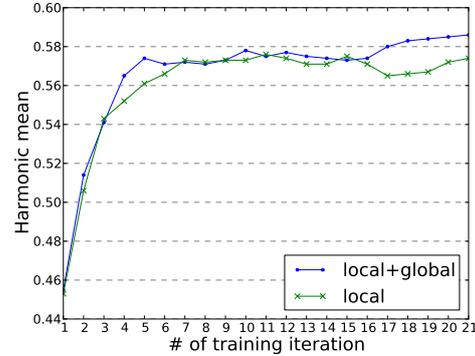


Figure 6: Training curves on dev set.

Figure 6 shows the training curves of the averaged perceptron with respect to the performance on the development set when the beam size is 4. As we can see both curves converge around iteration 20 and the global features improve the overall performance, compared to its counterpart with only local features. Therefore we set the number of iterations as 20 in the remaining experiments.

### 4.4 Impact of beam size

The beam size is an important hyper parameter in both training and test. Larger beam size will increase the computational cost while smaller beam size may reduce the performance. Table 4 shows the performance on the development set with several different beam sizes. When beam size = 4, the algorithm achieved the highest performance on the development set with trigger  $F_1 = 67.9$ , argument  $F_1 = 51.5$ , and harmonic mean = 58.6. When the size is increased to 32, the accuracy was not improved. Based on this observation, we chose beam size as 4 for the remaining experiments.

### 4.5 Early-update vs. standard-update

Huang et al. (2012) define “invalid update” to be an update that does not fix a violation (and instead reinforces the error), and show that it strongly (anti-)correlates with search quality and learning quality. Figure 7 depicts the percentage of invalid updates in standard-update with and without global features, respectively. With global features, there are numerous invalid updates when the

Beam size	1	2	4	8	16	32
Training time (sec)	993	2,034	3,982	8,036	15,878	33,026
Harmonic mean	57.6	57.7	<b>58.6</b>	58.0	57.8	57.8

Table 4: Comparison of training time and accuracy on the dev set.

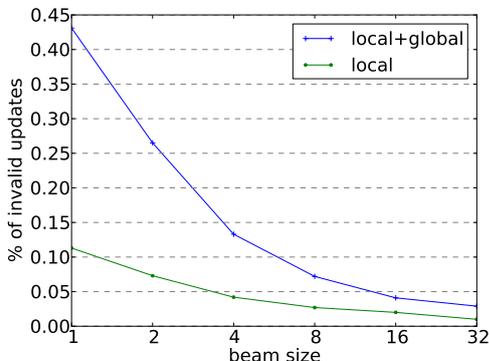


Figure 7: Percentage of the so-called “invalid updates” (Huang et al., 2012) in standard perceptron.

Strategy	F <sub>1</sub> on Dev		F <sub>1</sub> on Test	
	Trigger	Arg	Trigger	Arg
Standard ( $b = 1$ )	68.3	47.4	64.4	49.8
Early ( $b = 1$ )	<b>68.9</b>	<b>49.5</b>	<b>65.2</b>	<b>52.1</b>
Standard ( $b = 4$ )	<b>68.4</b>	50.5	67.1	51.4
Early ( $b = 4$ )	67.9	<b>51.5</b>	<b>67.5</b>	<b>52.7</b>

Table 5: Comparison between the performance (%) of standard-update and early-update with global features. Here  $b$  stands for beam size.

beam size is small. The ratio decreases monotonically as beam size increases. The model with only local features made much smaller numbers of invalid updates, which suggests that the use of global features makes the search problem much harder. This observation justifies the application of early-update in this work. To further investigate the difference between early-update and standard-update, we tested the performance of both strategies, which is summarized in Table 5. As we can see the performance of standard-update is generally worse than early-update. When the beam size is increased ( $b = 4$ ), the gap becomes smaller as the ratio of invalid updates is reduced.

#### 4.6 Overall performance

Table 6 shows the overall performance on the blind test set. In addition to our baseline, we compare against the sentence-level system reported in Hong et al. (2011), which, to the best of our knowledge,

is the best-reported system in the literature based on gold standard argument candidates. The proposed joint framework with local features achieves comparable performance for triggers and outperforms the staged baseline especially on arguments. By adding global features, the overall performance is further improved significantly. Compared to the staged baseline, it gains 1.6% improvement on trigger’s F-measure and 8.8% improvement on argument’s F-measure. Remarkably, compared to the cross-entity approach reported in (Hong et al., 2011), which attained 68.3% F<sub>1</sub> for triggers and 48.3% for arguments, our approach with global features achieves even better performance on argument labeling although we only used sentence-level information.

We also tested the performance with argument candidates automatically extracted by a high-performing name tagger (Li et al., 2012b) and an IE system (Grishman et al., 2005). The results are summarized in Table 7. The joint approach with global features significantly outperforms the baseline and the model with only local features. We also show that it outperforms the sentence-level baseline reported in (Ji and Grishman, 2008; Liao and Grishman, 2010), both of which attained 59.7% F<sub>1</sub> for triggers and 36.6% for arguments. Our approach aims to tackle the problem of sentence-level event extraction, thereby only using intra-sentential evidence. Nevertheless, the performance of our approach is still comparable with the best-reported methods based on cross-document and cross-event inference (Ji and Grishman, 2008; Liao and Grishman, 2010).

## 5 Related Work

Most recent studies about ACE event extraction rely on staged pipeline which consists of separate local classifiers for trigger labeling and argument labeling (Grishman et al., 2005; Ahn, 2006; Ji and Grishman, 2008; Chen and Ji, 2009; Liao and Grishman, 2010; Hong et al., 2011; Li et al., 2012a; Chen and Ng, 2012). To the best of our knowledge, our work is the first attempt to jointly model these two ACE event subtasks.

Methods	Trigger Identification (%)			Trigger Identification + classification (%)			Argument Identification (%)			Argument Role (%)		
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>
Sentence-level in Hong et al. (2011)	N/A			67.6	53.5	59.7	46.5	37.15	41.3	41.0	32.8	36.5
Staged MaxEnt classifiers	76.2	60.5	67.4	74.5	59.1	65.9	74.1	37.4	49.7	65.4	33.1	43.9
Joint w/ local features	77.4	62.3	69.0	73.7	59.3	65.7	69.7	39.6	50.5	64.1	36.5	46.5
Joint w/ local + global features	76.9	65.0	<b>70.4</b>	73.7	62.3	<b>67.5</b>	69.8	47.9	<b>56.8</b>	64.7	44.4	<b>52.7</b>
Cross-entity in Hong et al. (2011) <sup>†</sup>	N/A			72.9	64.3	68.3	53.4	52.9	53.1	51.6	45.5	48.3

Table 6: Overall performance with gold-standard entities, timex, and values. <sup>†</sup>beyond sentence level.

Methods	Trigger F <sub>1</sub>	Arg F <sub>1</sub>
Ji and Grishman (2008) cross-doc Inference	67.3	42.6
Ji and Grishman (2008) sentence-level	59.7	36.6
MaxEnt classifiers	64.7 (↓1.2)	33.7 (↓10.2)
Joint w/ local	63.7 (↓2.0)	35.8 (↓10.7)
Joint w/ local + global	<b>65.6 (↓1.9)</b>	<b>41.8 (↓10.9)</b>

Table 7: Overall performance (%) with predicted entities, timex, and values. ↓ indicates the performance drop from experiments with gold-standard argument candidates (see Table 6).

For the Message Understanding Conference (MUC) and FAS Program for Monitoring Emerging Diseases (ProMED) event extraction tasks, Patwardhan and Riloff (2009) proposed a probabilistic framework to extract event role fillers conditioned on the sentential event occurrence. Besides having different task definitions, the key difference from our approach is that their role filler recognizer and sentential event recognizer are trained independently but combined in the test stage. Our experiments, however, have demonstrated that it is more advantageous to do *both* training and testing with joint inference.

There has been some previous work on joint modeling for biomedical events (Riedel and McCallum, 2011a; Riedel et al., 2009; McClosky et al., 2011; Riedel and McCallum, 2011b). (McClosky et al., 2011) is most closely related to our approach. They casted the problem of biomedical event extraction as a dependency parsing problem. The key assumption that event structure can be considered as trees is incompatible with ACE event extraction. In addition, they used a separate classifier to predict the event triggers before applying the parser, while we extract the triggers and argument jointly. Finally, the features in the parser are edge-factorized. To exploit global features,

they applied a MaxEnt-based global re-ranker. In comparison, our approach is a unified framework based on beam search, which allows us to exploit arbitrary global features efficiently.

## 6 Conclusions and Future Work

We presented a joint framework for ACE event extraction based on structured perceptron with inexact search. As opposed to traditional pipelined approaches, we re-defined the task as a structured prediction problem. The experiments proved that the perceptron with local features outperforms the staged baseline and the global features further improve the performance significantly, surpassing the current state-of-the-art by a large margin.

As shown in Table 7, the overall performance drops substantially when using predicted argument candidates. To improve the accuracy of end-to-end IE system, we plan to develop a complete joint framework to recognize entities together with event mentions for future work. Also we are interested in applying this framework to other IE tasks such as relation extraction.

## Acknowledgments

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), U.S. NSF CAREER Award under Grant IIS-0953149, U.S. NSF EAGER Award under Grant No. IIS-1144111, U.S. DARPA Award No. FA8750-13-2-0041 in the “Deep Exploration and Filtering of Text” (DEFT) Program, a CUNY Junior Faculty Award, and Queens College equipment funds. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## References

- David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8.
- Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.
- Zheng Chen and Heng Ji. 2009. Language specific issue and feature exploration in chinese event extraction. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 209–212.
- Chen Chen and Vincent Ng. 2012. Joint modeling for chinese event extraction with rich linguistic features. In *COLING*, pages 529–544.
- Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 111.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8.
- Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.
- Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyu’s english ace 2005 system description. In *Proceedings of ACE 2005 Evaluation Workshop*. Washington.
- Yu Hong, Jianfeng Zhang, Bin Ma, Jian-Min Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of ACL*, pages 1127–1136.
- Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151.
- Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL*, pages 254–262.
- Peifeng Li, Guodong Zhou, Qiaoming Zhu, and Libin Hou. 2012a. Employing compositional semantics and discourse consistency in chinese event extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1006–1016.
- Qi Li, Haibo Li, Heng Ji, Wen Wang, Jing Zheng, and Fei Huang. 2012b. Joint bilingual name tagging for parallel corpora. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1727–1731.
- Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of ACL*, pages 789–797.
- Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. 1998. Nomlex: A lexicon of nominalizations. In *Proceedings of EU-RALEX*, volume 98, pages 187–193.
- David McClosky, Mihai Surdeanu, and Christopher D. Manning. 2011. Event extraction as dependency parsing. In *Proceedings of ACL*, pages 1626–1635.
- Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT-NAACL*, volume 4, pages 337–342.
- Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 151–160.
- Sebastian Riedel and Andrew McCallum. 2011a. Fast and robust joint models for biomedical event extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1–12.
- Sebastian Riedel and Andrew McCallum. 2011b. Robust biomedical event extraction with dual decomposition and minimal domain adaptation. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 46–50.
- Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun’ichi Tsujii. 2009. A markov logic approach to bio-molecular event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, pages 41–49.
- Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. Semi-supervised relation extraction with large-scale word clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 521–529.

# Language-Independent Discriminative Parsing of Temporal Expressions

**Gabor Angeli**  
Stanford University  
Stanford, CA 94305  
angeli@stanford.edu

**Jakob Uszkoreit**  
Google, Inc.  
1600 Amphitheatre Parkway  
Mountain View, CA 94303  
uszkoreit@google.com

## Abstract

Temporal resolution systems are traditionally tuned to a particular language, requiring significant human effort to translate them to new languages. We present a language independent semantic parser for learning the interpretation of temporal phrases given only a corpus of utterances and the times they reference. We make use of a latent parse that encodes a language-flexible representation of time, and extract rich features over both the parse and associated temporal semantics. The parameters of the model are learned using a weakly supervised bootstrapping approach, without the need for manually tuned parameters or any other language expertise. We achieve state-of-the-art accuracy on all languages in the TempEval-2 temporal normalization task, reporting a 4% improvement in both English and Spanish accuracy, and to our knowledge the first results for four other languages.

## 1 Introduction

Temporal resolution is the task of mapping from a textual phrase describing a potentially complex time, date, or duration to a normalized (*grounded*) temporal representation. For example, possibly complex phrases such as *the week before last*<sup>1</sup> are often more useful in their grounded form – e.g., August 4 – August 11.

Many approaches to this problem make use of rule-based methods, combining regular-expression matching and hand-written interpretation functions. In contrast, we would like to learn the interpretation of a temporal expression probabilistically. This allows propagation of uncertainty to higher-level components, and the potential to

dynamically back off to a rule-based system in the case of low confidence parses. In addition, we would like to use a representation of time which is broadly applicable to multiple languages, without the need for language-specific rules or manually tuned parameters.

Our system requires annotated data consisting only of an input phrase and an associated *grounded* time, relative to some reference time; the language-flexible parse is entirely latent. Training data of this weakly-supervised form is generally easier to collect than the alternative of manually creating and tuning potentially complex interpretation rules.

A large number of languages conceptualize time as lying on a one dimensional line. Although the surface forms of temporal expressions differ, the basic operations many languages use can be mapped to operations on this time line (see Section 3). Furthermore, many common languages share temporal units (hours, weekdays, etc.). By structuring a latent parse to reflect these semantics, we can define a single model which performs well on multiple languages.

A discriminative parsing model allows us to define sparse features over not only lexical cues but also the temporal value of our prediction. For example, it allows us to learn that we are much more likely to express *March 14<sup>th</sup>* than *2pm in March* – despite the fact that both interpretations are composed of similar types of components. Furthermore, it allows us to define both sparse n-gram and denser but less informative bag-of-words features over multi-word phrases, and allows us to handle numbers in a flexible way.

We briefly describe our temporal representation and grammar, followed by a description of the learning algorithm; we conclude with experimental results on the six languages of the TempEval-2 A task.

<sup>1</sup>Spoken on, for instance, August 20.

## 2 Related Work

Our approach follows the work of Angeli et al. (2012), both in the bootstrapping training methodology and the temporal grammar. Our foremost contributions over this prior work are: (i) the utilization of a discriminative parser trained with rich features; (ii) simplifications to the temporal grammar which nonetheless maintain high accuracy; and (iii) experimental results on 6 different languages, with state-of-the-art performance on both datasets on which we know of prior work.

As in this previous work, our approach draws inspiration from work on semantic parsing. The latent parse parallels the formal semantics in previous work. Supervised approaches to semantic parsing prominently include Zelle and Mooney (1996), Zettlemoyer and Collins (2005), Kate et al. (2005), Zettlemoyer and Collins (2007), *inter alia*. For example, Zettlemoyer and Collins (2007) learn a mapping from textual queries to a logical form. Importantly, the logical form of these parses contain all of the predicates and entities used in the parse – unlike the label provided in our case, where a grounded time can correspond to any of a number of latent parses. Along this line, recent work by Clarke et al. (2010) and Liang et al. (2011) relax supervision to require only annotated answers rather than full logical forms.

Related work on interpreting temporal expressions has focused on constructing hand-crafted interpretation rules (Mani and Wilson, 2000; Saquete et al., 2003; Puscasu, 2004; Grover et al., 2010). Of these, HeidelTime (Strötgen and Gertz, 2010) and SUTime (Chang and Manning, 2012) provide a strong comparison in English.

Recent probabilistic approaches to temporal resolution include UzZaman and Allen (2010), who employ a parser to produce deep logical forms, in conjunction with a CRF classifier. In a similar vein, Kolomiyets and Moens (2010) employ a maximum entropy classifier to detect the location and temporal type of expressions; the grounding is then done via deterministic rules.

In addition, there has been work on parsing Spanish expressions; UC3M (Vicente-Díez et al., 2010) produce the strongest results on the TempEval-2 corpus. Of the systems entered in the original task, TIPSem (Llorens et al., 2010) was the only system to perform bilingual interpretation for English and Spanish. Both of the above systems rely primarily on hand-built rules.

## 3 Temporal Representation

We define a compositional representation of time, similar to Angeli et al. (2012), but with a greater focus on efficiency and simplicity. The representation makes use of a notion of temporal *types* and their associated semantic *values*; a grammar is constructed over these types, and is grounded by appealing to the associated values.

A summary of the temporal type system is provided in Section 3.1; the grammar is described in Section 3.2; key modifications from previous work are highlighted in Section 3.3.

### 3.1 Temporal Types

Temporal expressions are represented either as a Range, Sequence, or Duration. The root of a parse tree should be one of these types. In addition, phrases can be tagged as a Function; or, as a special Nil type corresponding to segments without a direct temporal interpretation. Lastly, a type is allocated for numbers. We describe each of these briefly below.

**Range [and Instant]** A period between two dates (or times), as per an interval-based theory of time (Allen, 1981). This includes entities such as *Today*, *1987*, or *Now*.

**Sequence** A sequence of Ranges, occurring at regular but not necessarily constant intervals. This includes entities such as *Friday*, *November 27<sup>th</sup>*, or *last Friday*. A Sequence is defined in terms of a partial completion of calendar fields. For example, *November 27<sup>th</sup>* would define a Sequence whose year is unspecified, month is November, and day is the 27<sup>th</sup>; spanning the entire range of the lower order fields (in this case, a day). This example is illustrated in Figure 1. Note that a Sequence implicitly selects a possibly infinite number of possible Ranges.

To select a particular grounded time for a Sequence, we appeal to a notion of a *reference time* (Reichenbach, 1947). For the TempEval-2 corpus, we approximate this as the publication time of the article. While this is conflating Reichenbach’s reference time with speech time, and comes at the expense of certain mistakes (see Section 5.3), it is nonetheless useful in practice.

To a first approximation, grounding a sequence given a reference time corresponds to filling in the unspecified fields of the sequence with the fully-specified fields of the reference time. This pro-

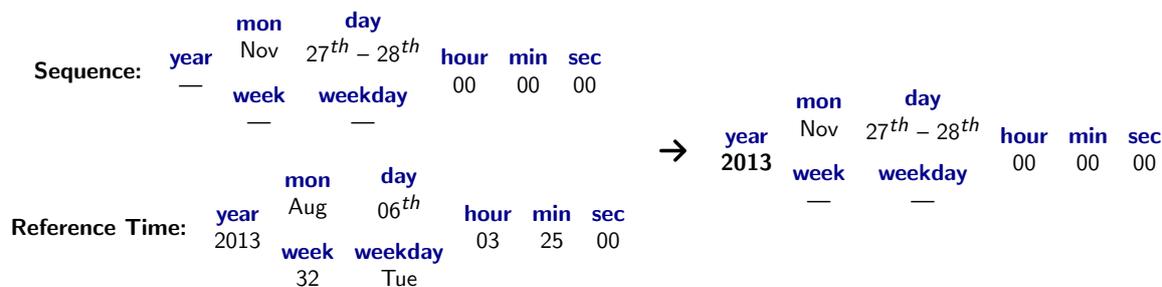


Figure 1: An illustration of grounding a Sequence. When grounding the Sequence November 27<sup>th</sup> with a reference time 2013-08-06 03:25:00, we complete the missing fields in the Sequence (the year) with the corresponding field in the reference time (2013).

cess has a number of special cases not enumerated here,<sup>2</sup> but the complexity remains constant time.

**Duration** A period of time. This includes entities like *Week*, *Month*, and *7 days*. A special case of the Duration type is defined to represent *approximate* durations, such as *a few years* or *some days*.

**Function** A function of arity less than or equal to two representing some general modification to one of the above types. This captures semantic entities such as those implied in *last x*, *the third x [of y]*, or *x days ago*. The particular functions are enumerated in Table 2.

**Nil** A special Nil type denotes terms which are not directly contributing to the semantic meaning of the expression. This is intended for words such as *a* or *the*, which serve as cues without bearing temporal content themselves.

**Number** Lastly, a special Number type is defined for tagging numeric expressions.

### 3.2 Temporal Grammar

Our approach assumes that natural language descriptions of time are compositional in nature; that is, each word attached to a temporal phrase is compositionally modifying the meaning of the phrase. We define a grammar jointly over temporal *types* and *values*. The types serve to constrain the parse and allow for coarse features; the values encode specific semantics, and allow for finer features. At the root of a parse tree, we recursively apply

<sup>2</sup>Some of these special cases are caused by variable days of the month, daylight savings time, etc. Another class arises from pragmatically peculiar utterances; e.g., *the next Monday in August* uttered in the last week of August should ground to August of next year (rather than the reference time’s year).

the functions in the tree to obtain a final temporal value.

This approach can be presented as a rule-to-rule translation (Bach, 1976; Allen, 1995, p. 263), or a constrained Synchronous PCFG (Yamada and Knight, 2001).

Formally, we define our grammar as  $G = (\Sigma, S, \mathcal{V}, T, \mathcal{R})$ . The alphabet  $\Sigma$  and start symbol  $S$  retain their usual interpretations. We define a set  $\mathcal{V}$  to be the set of types, as described in Section 3.1. For each  $v \in \mathcal{V}$  we define an (infinite) set  $T_v$  corresponding to the possible instances of type  $v$ . Each node in the tree defines a pair  $(v, \tau)$  such that  $\tau \in T_v$ . A rule  $R \in \mathcal{R}$  is defined as a pair  $R = (v_i \rightarrow v_j v_k, f : (T_{v_j}, T_{v_k}) \rightarrow T_{v_i})$ . This definition is trivially adapted for the case of unary rules.

The form of our rules reveals the synchronous aspect of our grammar. The structure of the tree is bound by the first part over types  $v$  – these types are used to populate the chart, and allow for efficient inference. The second part is used to evaluate the semantics of the parse,  $\tau \in T_{v_i}$ , and allows partial derivations to be discriminated based on richer information than the coarse types.

We adopt the preterminals of Angeli et al. (2012). Each preterminal consists of a *type* and a *value*; neither which are lexically informed. That is, the word *week* and preterminal (*Week*, *Duration*) are not tied in any way. A total of 62 preterminals are defined corresponding to instances of Ranges, Sequences, and Durations; these are summarized in Table 1.

In addition, 10 functions are defined for manipulating temporal expressions (see Table 2). The majority of these mirror generic operations on intervals on a timeline, or manipulations of a sequence. Notably, like intervals, times can be

Type	Example Instances
Range	Past, Future, Yesterday, Tomorrow, Today, Reference, Year ( $n$ ), Century ( $n$ )
Sequence	Friday, January, ... DayOfMonth, DayOfWeek, ... EveryDay, EveryWeek, ...
Duration	Second, Minute, Hour, Day, Week, Month, Quarter, Year, Decade, Century

Table 1: The content-bearing preterminals of the grammar, arranged by their types. Note that the Sequence type contains more elements than enumerated here; however, only a few of each characteristic type are shown here for brevity.

Function	Description
shiftLeft	Shift a Range left by a Duration
shiftRight	Shift a Range right by a Duration
shrinkBegin	Take the first Duration of a Range
shrinkEnd	Take the last Duration of a Range
catLeft	Take the Duration after a Range
catRight	Take the Duration before a Range
moveLeft1	Shift a Sequence left by 1
moveRight1	Shift a Sequence right by 1
$n^{\text{th}}$ $x$ of $y$	Take the $n^{\text{th}}$ element in $y$
approximate	Make a Duration approximate

Table 2: The functional preterminals of the grammar. The name and a brief description of the function are given; the functions are most easily interpreted as operations on either an interval or sequence. All operations on Ranges can equivalently be applied to Sequences.

moved (*3 weeks ago*) or their size changed (*the first two days of the month*), or a new interval can be started from one of the endpoints (*the last 2 days*). Additionally, a sequence can be modified by shifting its origin (*last Friday*), or taking the  $n^{\text{th}}$  element of the sequence within some bound (*fourth Sunday in November*).

Combination rules in the grammar mirror type-checked curried function application. For instance, the function `moveLeft1` applied to `week` (as in *last week*) yields a grammar rule:

$$\begin{array}{c}
 (\text{EveryWeek } -1, \text{Seq.}) \\
 \swarrow \quad \searrow \\
 (\text{moveLeft1}, \text{Seq.} \rightarrow \text{Seq.}) \quad (\text{EveryWeek}, \text{Seq.})
 \end{array}$$

In more generality, we create grammar rules for applying a function on either the left or the right, for all possible type signatures of  $f: f(x, y) \odot x$  or  $x \odot f(x, y)$ .

Additionally, a grammar rule is created for intersecting two Ranges or Sequences, for multiplying a duration by a number, and for absorbing a Nil span. Each of these can be thought of as an implicit function application (in the last case, the identity function).

### 3.3 Differences From Previous Work

While the grammar formalism is strongly inspired by Angeli et al. (2012), a number of key differences are implemented to both simplify the framework, and make inference more efficient.

**Sequence Grounding** The most time-consuming and conceptually nuanced aspect of temporal inference in Angeli et al. (2012) is intersecting Sequences. In particular, there are two modes of expressing dates which resist intersection: a day-of-month-based mode and a week-based mode. Properly grounding a sequence which defines both a day of the month and a day of the week (or week of the year) requires backing off to an expensive search problem.

To illustrate, consider the example: *Friday the 13<sup>th</sup>*. Although both a Friday and a 13<sup>th</sup> of the month are easily found, the intersection of the two requires iterating through elements of one until it overlaps with an element of the other. At training time, a number of candidate parses are generated for each phrase. When considering that these parses can become both complex and pragmatically unreasonable, this can result in a noticeable efficiency hit; e.g., during training a sentence could have a [likely incorrect] candidate interpretation of: *nineteen ninety-six Friday the 13<sup>th</sup>s from now*.

In our Sequence representation, such intersections are disallowed, in the same fashion as February 30<sup>th</sup> would be.

**Sequence Pragmatics** For the sake of simplicity the pragmatic distribution over possible groundings of a sequence is replaced with the single most likely offset, as learned empirically from the English TempEval-2 corpus by Angeli et al. (2012).

**No Tag Splitting** The Number and Nil types are no longer split according to their ordinality/magnitude and subsumed phrase, respectively.

More precisely, there is a single nonterminal (Nil), rather than a nonterminal symbol characterizing the phrase it is subsuming (Nil-*the*, Nil-*a*, etc.). This information is encoded more elegantly as features.

## 4 Learning

The system is trained using a discriminative  $k$ -best parser, which is able to incorporate arbitrary features over partial derivations. We describe the parser below, followed by the features implemented.

### 4.1 Parser

**Inference** A discriminative  $k$ -best parser was used to allow for arbitrary features in the parse tree. In the first stage, spans of the input sentence are tagged as either text or numbers. A rule-based number recognizer was used for each language to recognize and ground numeric expressions, including information on whether the number was an ordinal (e.g., *two* versus *second*). Note that unlike conventional parsing, a tag can span multiple words. Numeric expressions are treated as if the numeric value replaced the expression.

Each rule of the parse derivation was assigned a score according to a log-linear factor. Specifically, each rule  $R = (v_i \rightarrow v_j v_k, f)$  with features over the rule and derivation so far  $\phi(R)$ , subject to parameters  $\theta$ , is given a probability:

$$P(v_i | v_j, v_k, f; \theta) \propto e^{\theta^T \phi(R)} \quad (1)$$

Naïvely, this parsing algorithm gives us a complexity of  $O(n^3 k^2)$ , where  $n$  is the length of the sentence, and  $k$  is the size of the beam. However, we can approximate the algorithm in  $O(n^3 k \log k)$  time with cube pruning (Chiang, 2007). With features which are not context-free, we are not guaranteed an optimal beam with this approach; however, empirically the approximation yields a significant efficiency improvement without noticeable loss in performance.

**Training** We adopt an EM-style bootstrapping approach similar to Angeli et al. (2012), in order to handle the task of parsing the temporal expression without annotations for the latent parses. Each training instance is a tuple consisting of the words in the temporal phrase, the annotated grounded time  $\tau^*$ , and the reference time.

Given an input sentence, our parser will output  $k$  possible parses; when grounded to the

reference time these correspond to  $k$  candidate times:  $\tau_1 \dots \tau_k$ , each with a normalized probability  $P(\tau_i)$ . This corresponds to an approximate E step in the EM algorithm, where the distribution over latent parses is approximated by a beam of size  $k$ . Although for long sentences the number of parses is far greater than the beam size, as the parameters improve, increasingly longer sentences will have correct derivations in the beam. In this way, a progressively larger percentage of the data is available to be learned from at each iteration.

To approximate the M step, we define a multi-class hinge loss  $l(\theta)$  over the beam, and optimize using Stochastic Gradient Descent with AdaGrad (Duchi et al., 2010):

$$l(\theta) = \max_{0 \leq i < k} \mathbb{1}[\tau_i \neq \tau^*] + P_\theta(\tau_i) - P_\theta(\tau^*) \quad (2)$$

We proceed to describe our features.

### 4.2 Features

Our framework allows us to define arbitrary features over partial derivations. Importantly, this allows us to condition not only on the PCFG probabilities over *types* but also the partial semantics of the derivation. We describe the features used below; a summary of these features for a short phrase is illustrated in Figure 2.

**Bracketing Features** A feature is defined over every nonterminal combination, consisting of the pair of children being combined in that rule. In particular, let us consider a rule  $R = (v_i \rightarrow v_j v_k, f)$  corresponding to a CFG rule  $v_i \rightarrow v_j v_k$  over *types*, and a function  $f$  over the semantic values corresponding to  $v_j$  and  $v_k$ :  $\tau_j$  and  $\tau_k$ . Two classes of bracketing features are extracted: features are extracted over the types of nonterminals being combined ( $v_j$  and  $v_k$ ), and over the top-level semantic derivation of the nonterminals ( $f$ ,  $\tau_j$ , and  $\tau_k$ ).

Unlike syntactic parsing, child types of a parse tree uniquely define the parent type of the rule; this is a direct consequence of our combination rules being functions with domains defined in terms of the temporal types, and therefore necessarily projecting their inputs into a single output type. Therefore, the first class of bracketing features – over types – reduce to have the exact same expressive power as the nonterminal CFG rules of Angeli et al. (2012). Examples of features in this class are features 13 and 15 in Figure 2 (b).

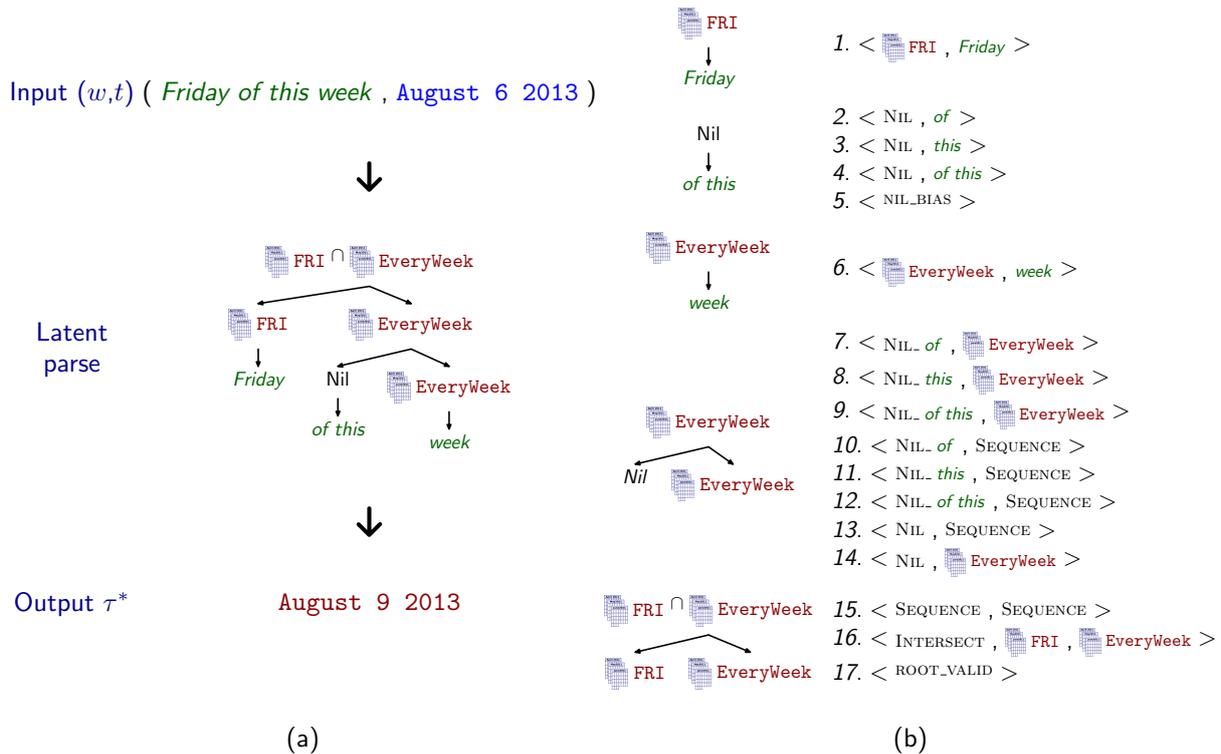


Figure 2: An example parse of *Friday of this week*, along with the features extracted from the parse. A summary of the input, latent parse, and output for a particular example is given in (a). The features extracted for each fragment of the parse are given in (b), and described in detail in Section 4.2.

We now also have the flexibility to extract a second class of features from the semantics of the derivation. We define a feature bracketing the most recent semantic function applied to each of the two child derivations; along with the function being applied in the rule application. If the child is a preterminal, the semantics of the preterminal are used; otherwise, the outermost (most recent) function to be applied to the derivation is used. To illustrate, a tree fragment combining *August* and *2013* into *August 2013* would yield the feature  $\langle \text{INTERSECT} , \text{AUGUST} , \text{2013} \rangle$ . This can be read as a feature for the rule applying the intersect function to *August* and *2013*. Furthermore, intersecting *August 2013* with the *12<sup>th</sup>* of the month would yield a feature  $\langle \text{INTERSECT} , \text{INTERSECT} , \text{12<sup>th</sup>} \rangle$ . This can be read as applying the intersect function to a subtree which is the intersection of two terms, and to the *12<sup>th</sup>* of the month. Features 14 and 16 in Figure 2 (b) are examples of such features.

**Lexical Features** The second large class of features extracted are lexicalized features. These are primarily used for tagging phrases with pretermi-

nals; however, they are also relevant in incorporating cues from the yield of Nil spans. To illustrate, *a week* and *the week* have very different meanings, despite differing by only their Nil tagged tokens.

In the first case, a feature is extracted over the *value* of the preterminal being extracted, and the phrase it is subsuming (e.g., features 1–4 and 6 in Figure 2 (b)). As the type of the preterminal is deterministic from the value, encoding a feature on the type of the preterminal would be a coarser encoding of the same information, and is empirically not useful in this case. Since a multi-word expression can parse to a single nonterminal, a feature is extracted for the entire  $n$ -gram in addition to features for each of the individual words. For example, the phrase *of this* – of type Nil – would have features extracted:  $\langle \text{NIL} , \text{of} \rangle$ ,  $\langle \text{NIL} , \text{this} \rangle$ , and  $\langle \text{NIL} , \text{of this} \rangle$ .

In the second case – absorbing Nil-tagged spans – we extract features over the words under the Nil span joined with the type and value of the other derivation (e.g., features 7–12 in Figure 2 (b)). As above, features are extracted for both  $n$ -grams and for each word in the phrase. For example, combining *of this* and *week* would yield features

System	Train		Test	
	Type	Value	Type	Value
GUTime	0.72	0.46	0.80	0.42
SUTime	0.85	0.69	<b>0.94</b>	0.71
HeidelTime	0.80	0.67	0.85	0.71
ParsingTime	0.90	0.72	0.88	0.72
OurSystem	<b>0.94</b>	<b>0.81</b>	0.91	<b>0.76</b>

Table 3: English results for TempEval-2 attribute scores for our system and four previous systems. The scores are calculated using gold extents, forcing an interpretation for each parse.

System	Train		Test	
	Type	Value	Type	Value
UC3M	—	—	0.79	0.72
OurSystem	0.90	0.84	<b>0.92</b>	<b>0.76</b>

Table 4: Spanish results for TempEval-2 attribute scores for our system and the best known previous system. The scores are calculated using gold extents, forcing an interpretation for each parse.

<of, EVERYWEEK>, <this, EVERYWEEK>, and <of this, EVERYWEEK>.

In both cases, numbers are featurized according to their order of magnitude, and whether they are ordinal. Thus, the number tagged from *thirty-first* would be featurized as an ordinal number of magnitude 2.

**Semantic Validity** Although some constraints can be imposed to help ensure that a top-level parse will be valid, absolute guarantees are difficult. For instance, February 30 is never a valid date; but, it would be difficult to disallow any local rule in its derivation. To mediate this, an indicator feature is extracted denoting whether the grounded semantics of the derivation is valid. This is illustrated in Figure 2 (b) by feature 17.

**Nil Bias** Lastly, an indicator feature is extracted for each Nil span tagged (feature 5 in Figure 2 (b)). In part, this discourages over-generation of the type; in another part, it encourages Nil spans to absorb as many adjacent words as possible.

We proceed to describe our experimental setup and results.

## 5 Evaluation

We evaluate our model on all six languages in the TempEval-2 Task A dataset (Verhagen et al.,

2010), comparing against state-of-the-art systems for English and Spanish. New results are reported on smaller datasets from the four other languages. To our knowledge, there has not been any prior work on these corpora.

We describe the TempEval-2 datasets in Section 5.1, present experimental results in Section 5.2, and discuss system errors in Section 5.3.

### 5.1 TempEval-2 Datasets

TempEval-2, from SemEval 2010, focused on retrieving and reasoning about temporal information from newswire. Our system evaluates against Task A – detecting and resolving temporal expressions. Since we perform only the second of these, we evaluate our system assuming gold detection.

The dataset annotates six languages: English, Spanish, Italian, French, Chinese, and Korean; of these, English and Spanish are the most mature. We describe each of these languages, along with relevant quirks, below:

**English** The English dataset consists of 1052 training examples, and 156 test examples. Evaluation was done using the official evaluation script, which checks for exact match between `TIMEX3` tags. Note that this is stricter than our training objective; for instance, *24 hours* and *a day* have the same interpretation, but have different `TIMEX3` strings. System output was heuristically converted to the `TIMEX3` format; where ambiguities arose, the convention which maximized training accuracy was chosen.

**Spanish** The Spanish dataset consists of 1092 training examples, and 198 test examples. Evaluation was identical to the English, with the heuristic `TIMEX3` conversion adapted somewhat.

**Italian** The Italian dataset consists of 523 training examples, and 126 test examples. Evaluation was identical to English and Spanish.

**Chinese** The Chinese dataset consists of 744 training examples, and 190 test examples. Of these, only 659 training and 143 test examples had a temporal value marked; the remaining examples had a type but no value, and are therefore impossible to predict. Results are also reported on a clean corpus with these impossible examples omitted.

The Chinese, Korean, and French corpora had noticeable inconsistencies in the `TIMEX3` annotation. Thus, evaluations are reported according

Language	Train			Test		
	# examples	Type	Value	# examples	Type	Value
English	1052	0.94	0.81	156	0.91	0.76
Spanish	1092	0.90	0.84	198	0.92	0.76
Italian	523	0.89	0.85	126	0.84	0.38
Chinese <sup>†</sup>	744	0.95	0.65	190	0.87	0.48
Chinese (clean) <sup>†</sup>	659	0.97	0.73	143	0.97	0.60
Korean <sup>†</sup>	247	0.83	0.67	91	0.82	0.42
French <sup>†</sup>	206	0.78	0.76	83	0.78	0.35

Table 5: Our system’s accuracy on all 6 languages of the TempEval-2 corpus. Chinese is divided into two results: one for the entire corpus, and one which considers only examples for which a temporal value is annotated. Languages with a dagger (<sup>†</sup>) were evaluated based on semantic rather than string-match correctness.

to the training objective: if two `TIMEX3` values ground to the same grounded time, they are considered equal. For example, in the example above, *24 hours* and *a day* would be marked identical despite having different `TIMEX3` strings.

Most `TIMEX3` values convert naturally to a grounded representation; values with wildcards representing Sequences (e.g., `1998-QX` or `1998-XX-12`) ground to the same value as the Sequence encoding that value would. For instance, `1998-QX` is parsed as *every quarter in 1998*.

**Korean** The Korean dataset consists of 287 training examples, and 91 test examples. 40 of the training examples encoded dates as a long integer. For example: `003000000200001131951006` grounds to January 13, 2000 at the time 19:51. These were removed from the training set, yielding 247 examples; however, all three such examples were left in the test set. Evaluation was done identically to the Chinese data.

**French** Lastly, a dataset for French temporal expressions was compiled from the TempEval-2 data. Unlike the other 5 languages, the French data included only the raw `TIMEX3` annotated newswire documents, encoded as XML. These documents were scraped to recover 206 training examples and 83 test examples. Evaluation was done identically to the Chinese and Korean data.

We proceed to describe our experimental results on these datasets.

## 5.2 Results

We compare our system with state-of-the-art systems for both English and Spanish. To the best of our knowledge, no prior work exists for the other

four languages.

We evaluate in the same framework as Angeli et al. (2012). We compare to previous system scores when constrained to make a prediction on every example; if no guess is made, the output is considered incorrect. This in general yields lower results for those systems, as the system is not allowed to abstain on expressions it does not recognize.

The systems compared against are:

- GUTime (Mani and Wilson, 2000), a widely used, older rule-based system.
- HeidelTime (Strötgen and Gertz, 2010), the top system at the TempEval-2 task for English.
- SUTime (Chang and Manning, 2012), a more recent rule-based system for English.
- ParsingTime (Angeli et al., 2012), a semantic parser for temporal expressions, similar to this system (see Section 2).
- UC3M (Vicente-Díez et al., 2010), a rule-based system for Spanish.

Results for the English corpus are shown in Table 3. Results for Spanish are shown in Table 4. Lastly, a summary of results in all six languages is shown in Table 5.

A salient trend emerges from the results – while training accuracy is consistently high, test accuracy drops sharply for the data-impooverished languages. This is consistent with what would be expected from a discriminatively trained model in data-impooverished settings; however, the consistent training accuracy suggests that the model nonetheless captures the phenomena it sees in

Error Class	English	Spanish
Pragmatics	29%	23%
Type error	16%	5%
Incorrect number	10%	5%
Relative Range	7%	2%
Incorrect parse	19%	36%
Missing context	16%	23%
Bad reference time	3%	6%

Table 6: A summary of errors of our system, by percentage of incorrect examples for the English and Spanish datasets. The top section describes errors which could be handled in our framework, while the bottom section describes examples which are either ambiguous (missing context), or are annotated inconsistently relative the reference time.

training. This suggests the possibility for improving accuracy further by making use of more data during training.

### 5.3 Discussion

We characterize the examples our system parses incorrectly on the English and Spanish datasets in Table 6, expanding on each class of error below.

**Pragmatics** This class of errors is a result of pragmatic ambiguity over possible groundings of a sequence – for instance, it is often ambiguous whether *next weekend* refers to the coming or subsequent weekend. These errors manifest in either dropping a function (*next*, *last*), or imagining one that is not supported by the text (e.g., *this week* parsed as next week).

**Type error** Another large class of errors – particularly in the English dataset – arise from not matching the annotation’s type, but otherwise producing a reasonable response. For instance, the system may mistake a day on the calendar (a Range), with a day, the period of time.

**Incorrect number** A class of mistakes arises from either omitting numbers from the parse, or incorrectly parsing numbers – the second case is particularly prevalent for written years, such as *seventeen seventy-six*.

**Relative Range** These errors arise from attempting to parse a grounded Range by applying functions to the reference time. For example, from a reference time of August 8<sup>th</sup>, it is possible to

“correctly” parse the phrase *August 1* as a week ago; but, naturally, this parse does not generalize well. This class of errors, although relatively small, merits special designation as it suggests a class of correct responses which are correct for the wrong reasons. Future work could explore mitigating these errors for domains where the text is further removed from the events it is describing than most news stories are.

**Incorrect parse** Errors in this class are a result of failing to find the correct parse, for a number of reasons not individually identified. A small subset of these errors (notably, 6% on the Spanish dataset) are a result of the grammar being insufficiently expressive with the preterminals we defined. For instance, we cannot capture fractional units, such as in *half an hour*.

**Missing context** A fairly large percentage of our errors arise from failing to classify inputs which express ambiguous or poorly defined times. For example, *from time to time* (annotated as the future), or *that time* (annotated as 5 years). Many of these require either some sort of inference or a broader understanding of the context in which the temporal phrase is uttered, which our system does not attempt to capture.

**Bad reference time** The last class of errors cover cases where the temporal phrase is clear, but annotation differs from our judgment of what would be reasonable. These are a result of assuming that the reference time of an utterance is the publication time of the article.

## 6 Conclusion

We have presented a discriminative, multilingual approach to resolving temporal expressions, using a language-flexible latent parse and rich features on both the *types* and *values* of partial derivations in the parse. We showed state-of-the-art results on both languages in TempEval-2 with prior work, and presented results on four additional languages.

**Acknowledgments** Work was done in the summer of 2012 while the first author was an intern at Google. We would like to thank Chris Manning, and our co-workers at Google for their insight and help.

## References

- James F. Allen. 1981. An interval-based representation of temporal knowledge. In *Proceedings of the 7th international joint conference on Artificial intelligence*, pages 221–226, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- James Allen. 1995. *Natural Language Understanding*. Benjamin/Cummings, Redwood City, CA.
- Gabor Angeli, Christopher D. Manning, and Daniel Jurafsky. 2012. Parsing time: Learning to interpret time expressions. In *NAACL-HLT*.
- E. Bach. 1976. An extension of classical transformational grammar. In *Problems of Linguistic Metatheory (Proceedings of the 1976 Conference)*, Michigan State University.
- Angel Chang and Chris Manning. 2012. SUTIME: a library for recognizing and normalizing time expressions. In *Language Resources and Evaluation*.
- David Chiang. 2007. Hierarchical phrase-based translation. *computational linguistics*, 33(2):201–228.
- James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world’s response. In *CoNLL*, pages 18–27, Uppsala, Sweden.
- John Duchi, Elad Hazan, and Yoram Singer. 2010. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.
- Claire Grover, Richard Tobin, Beatrice Alex, and Kate Byrne. 2010. Edinburgh-LTG: TempEval-2 system description. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, Sem-Eval, pages 333–336.
- Rohit J. Kate, Yuk Wah Wong, and Raymond J. Mooney. 2005. Learning to transform natural to formal languages. In *AAAI*, pages 1062–1068, Pittsburgh, PA.
- Oleksandr Kolomiyets and Marie-Francine Moens. 2010. KUL: recognition and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, Sem-Eval ’10, pages 325–328.
- P. Liang, M. I. Jordan, and D. Klein. 2011. Learning dependency-based compositional semantics. In *ACL*.
- Hector Llorens, Estela Saquete, and Borja Navarro. 2010. Tipsem (english and spanish): Evaluating crfs and semantic roles in tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 284–291.
- Inderjeet Mani and George Wilson. 2000. Robust temporal processing of news. In *ACL*, pages 69–76, Hong Kong.
- G. Puscasu. 2004. A framework for temporal resolution. In *LREC*, pages 1901–1904.
- Hans Reichenbach. 1947. *Elements of Symbolic Logic*. Macmillan, New York.
- E. Saquete, R. Muoz, and P. Martnez-Barco. 2003. Terseo: Temporal expression resolution system applied to event ordering. In *Text, Speech and Dialogue*, pages 220–228.
- Jannik Strötgen and Michael Gertz. 2010. Heildeltime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, Sem-Eval, pages 321–324.
- Naushad UzZaman and James F. Allen. 2010. TRIPS and TRIOS system for TempEval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, Sem-Eval, pages 276–283.
- Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. Semeval-2010 task 13: TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 57–62, Uppsala, Sweden.
- María Teresa Vicente-Díez, Julián Moreno Schneider, and Paloma Martínez. 2010. Uc3m system: Determining the extent, type and value of time expressions in tempeval-2. In *proceedings of the Semantic Evaluation–2 (Semeval 2010), ACL Conference, Uppsala (Sweden)*.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *ACL*, pages 523–530.
- John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI*, pages 1050–1055, Portland, OR.
- Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorical grammars. In *UAI*, pages 658–666. AUAI Press.
- Luke S. Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *EMNLP-CoNLL*, pages 678–687.

# Graph-based Local Coherence Modeling

Camille Guinaudeau and Michael Strube

Heidelberg Institute for Theoretical Studies gGmbH

Schloss-Wolfsbrunnenweg 35

69118 Heidelberg, Germany

(camille.guinaudeau|michael.strube)@h-its.org

## Abstract

We propose a computationally efficient graph-based approach for local coherence modeling. We evaluate our system on three tasks: sentence ordering, summary coherence rating and readability assessment. The performance is comparable to entity grid based approaches though these rely on a computationally expensive training phase and face data sparsity problems.

## 1 Introduction

Many NLP applications which process or generate texts rely on information about local coherence, i.e. information about which entities occur in which sentence and how the entities are distributed in the text. This led to the development of many theories and models accounting for local coherence. One popular model, the centering model (Grosz et al., 1995), uses a ranking of discourse entities realized in particular sentences and computes transitions between adjacent sentences to provide insight in the felicity of texts. Centering models local coherence rather generally and has been applied to the generation of referring expressions (Kibble and Power, 2004), to resolve pronouns (Brennan et al., 1987, inter alia), to score essays (Miltakaki and Kukich, 2004), to arrange sentences in the correct order (Karamanis et al., 2009), and to many other tasks. Poesio et al. (2004) observe that it is not clear how to set parameters in the centering model so that optimal performance in different tasks and languages can be achieved. Barzilay and Lapata (2008) criticize research on centering to be too dependent on manually annotated input. This led them to propose a local coherence model relying on a more parsimonious representation, the entity grid model.

The entity grid is a two dimensional array where the rows represent sentences and the columns discourse entities. From this grid Barzilay and Lapata (2008) derive probabilities of transitions between adjacent sentences which are used as features for machine learning algorithms. They evaluate this approach successfully on sentence ordering, summary coherence rating, and readability assessment. However, their approach has some disadvantages which they point out themselves: data sparsity, domain dependence and computational complexity, especially in terms of feature space issues while building their model (Barzilay and Lapata (2008, p.8, p.10, p.30), Elsner and Charniak (2011, p.126, p.127)).

In order to overcome these problems we propose to represent entities in a graph and then model local coherence by applying centrality measures to the nodes in the graph (Section 3). We claim that a graph is a more powerful representation for local coherence than the entity grid (Barzilay and Lapata, 2008) which is restricted to transitions between adjacent sentences. The graph can easily span the entire text without leading to computational complexity and data sparsity problems. Similar to the application of graph-based methods in other areas of NLP (e.g. work on word sense disambiguation by Navigli and Lapata (2010); for an overview over graph-based methods in NLP see Mihalcea and Radev (2011)) we model local coherence by relying only on centrality measures applied to the nodes in the graph. We apply our graph-based model to the three tasks handled by Barzilay and Lapata (2008) to show that it provides the same flexibility over disparate tasks as the entity grid model: sentence ordering (Section 4.1), summary coherence ranking (Section 4.2), and readability assessment (Section 4.3). In the

The Turkish government fell after mob-tie allegations.

Turkey’s constitution mandates a secular republic despite its Muslim majority.

Military and secular leaders pressured President Demirel to keep the Islamic-oriented Virtue Party on the fringe.

Business leaders feared Virtue would alienate the EU.

Table 1: Excerpt of a manual summary M from DUC2003

experiments sections, we discuss the impact of genre and stylistic properties of documents on the local coherence computation. We also show that, though we do not need a computationally expensive learning phase, our model achieves state-of-the-art performance. From this we conclude that a graph is an alternative to the entity grid model: it is computationally more tractable for modeling local coherence and does not suffer from data sparsity problems (Section 5).

## 2 The Entity Grid Model

Barzilay and Lapata (2005; 2008) introduced the entity grid, a method for local coherence modeling that captures the distribution of discourse entities across sentences in a text.

An entity grid is a two dimensional array, where rows correspond to sentences and columns to discourse entities. For each discourse entity  $e_j$  and each sentence  $s_i$  in the text, the corresponding grid cell  $c_{ij}$  contains information about the presence or absence of the entity in the sentence. If the entity does not appear in the sentence, the corresponding grid cell contains an absence marker “-”. If the entity is present in the sentence, the cell contains a representation of the entity’s syntactic role: “S” if the entity is a subject, “O” if it is an object and “X” for all other syntactic roles (cf. Table 2). When a noun is attested more than once with a different grammatical role in the same sentence, the role with the highest grammatical ranking is chosen to represent the entity (a subject is ranked higher than an object, which is ranked higher than other syntactic roles).

Barzilay and Lapata (2008) capture local coherence by means of local entity transitions, i.e. sequences of grid cells  $(c_{1j} \dots c_{ij} \dots c_{nj})$  representing the syntactic function or absence of an entity in adjacent sentences<sup>1</sup>. The coherence of a sentence in relation to its local context is determined by the

<sup>1</sup>For complexity reasons, Barzilay and Lapata consider only transitions between at most three sentences.

	GOVERNMENT	ALLEGATION	TURKEY	CONSTITUTION	SECULAR	REPUBLIC	MAJORITY	MILITARY	LEADER	PRESIDENT	DEMIREL	VIRTUE	PARTY	FRINGE	BUSINESS	EU
$s_1$	S	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-
$s_2$	-	-	X	S	X	O	X	-	-	-	-	-	-	-	-	-
$s_3$	-	-	-	-	X	-	-	X	S	X	S	X	O	X	-	-
$s_4$	-	-	-	-	-	-	-	-	S	-	-	S	-	-	X	O

Table 2: Entity Grid representation of summary M

local entity transitions of the entities present or absent in the sentence. To make this representation accessible to machine learning algorithms, Barzilay and Lapata (2008) compute for each document the probability of each transition and generate feature vectors representing the sentences. Coherence assessment is then formulated as a ranking learning problem where the ranking function is learned with SVM<sup>light</sup> (Joachims, 2002).

The entity grid approach has already been applied to many applications relying on local coherence estimation: summary rating (Barzilay and Lapata, 2005), essay scoring (Burstein et al., 2010) or story generation (McIntyre and Lapata, 2010). It was also used successfully in combination with other systems or features. Soricut and Marcu (2006) show that the entity grid model is a critical component in their sentence ordering model for discourse generation. Barzilay and Lapata (2008) combine the entity grid with readability-related features to discriminate documents between easy- and difficult-to-read categories. Lin et al. (2011) use discourse relations to transform the entity grid representation into a discourse role matrix that is used to generate feature vectors for machine learning algorithms similarly to Barzilay and Lapata (2008).

Several studies propose to extend the entity grid model using different strategies for entity selection. Filippova and Strube (2007) aim to improve the entity grid model performance by grouping entities by means of semantic relatedness. In their studies, Elsner and Charniak extend the number and type of entities selected and consider that each entity has to be dealt with accordingly with its information status (Elsner et al., 2007) or its named-entity category (Elsner and Charniak, 2011). Finally, they include a heuristic coreference resolution component by linking mentions which share a

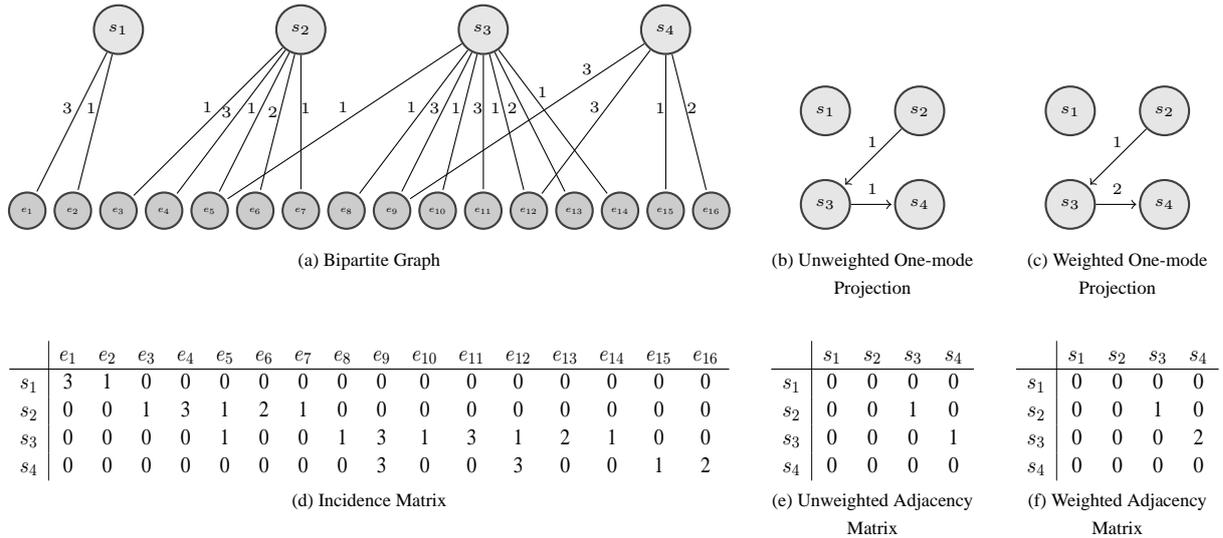


Figure 1: Bipartite graph for summary M from Table 1, one-mode projections and associated incidence and adjacency matrices. Weights in Figure 1(a) are assigned as follows: “S” = 3, “O” = 2, “X” = 1, “-” = 0 (no edge).

head noun. These extensions led to the best results reported so far for the sentence ordering task.

### 3 Method

Our model is based on the insight that the entity grid (Barzilay and Lapata, 2008) corresponds to the incidence matrix of a bipartite graph representing the text (see Newman (2010) for more details on graph representation). A fundamental assumption underlying our model is that this bipartite graph contains the entity transition information needed for local coherence computation, rendering feature vectors and learning phase unnecessary. The bipartite graph  $G = (V_s, V_e, L, w)$  is defined by two independent sets of nodes – that correspond to the set of sentences  $V_s$  and the set of entities  $V_e$  of the text – and a set of edges  $L$  associated with weights  $w$ . An edge between a sentence node  $s_i$  and an entity node  $e_j$  is created in the bipartite graph if the corresponding cell  $c_{ij}$  in the entity grid is not equal to “-”. Each edge is associated with a weight  $w(e_j, s_i)$  that depends on the grammatical role of the entity  $e_j$  in the sentence  $s_i$ <sup>2</sup>. In contrast to Barzilay and Lapata’s entity grid that contains information about absent entities, our graph-based representation only contains “positive” information. Figure 1(a) shows an example of the bipartite graph that corresponds to the grid in Table 2. The incidence matrix of this graph (Figure 1(d)) is very similar to the entity grid.

<sup>2</sup>The assignment of weights is described in Section 4.

By modeling entity transitions, Barzilay and Lapata rely on links that exist between sentences to model local coherence. In the same spirit, we apply different kinds of one-mode projections to the sentence node set  $V_s$  of the bipartite graph to represent the connections that exist between – potentially non adjacent – sentences in the graph. These projections result in graphs where nodes correspond to sentences. An edge is created between two nodes if the corresponding sentences have a least one entity in common. Contrary to the bipartite graph, one-mode projections are directed as they follow the text order. Therefore, in projection graphs an edge can exist between the first and the second sentence while the inverse is not possible. In our model, we define three kinds of projection graphs,  $P_U$ ,  $P_W$  and  $P_{Acc}$ , depending on the weighting scheme associated with their edges. In  $P_U$ , weights are binary and equal 1 when two sentences have a least one entity in common (Figure 1(b)). In  $P_W$ , edges are weighted according to the number of entities “shared” by two sentences (Figure 1(c)). In  $P_{Acc}$  syntactic information is accounted for by integrating the edge weights in the bipartite graph. In this case, weights are equal to

$$W_{ik} = \sum_{e \in E_{ik}} w(e, s_i) \cdot w(e, s_k),$$

where  $E_{ik}$  is the set of entities shared by  $s_i$  and  $s_k$ . Distance between sentences  $s_i$  and  $s_k$  can also be integrated in the weight of one-mode projections to decrease the importance of links that ex-

ists between non adjacent sentences. In this case, the weights of the projection graphs are divided by  $k - i$ .

From this graph-based representation, the local coherence of a text  $T$  can be measured by computing the average outdegree of a projection graph  $P$ . This centrality measure was chosen for two main reasons. First, it allows us to evaluate to which extent a sentence is connected, in terms of discourse entities, with the other sentences of the text. Second, compared to other centrality measures, the computational complexity of the average outdegree is low ( $\mathcal{O}(\frac{N*(N-1)}{2})$  for a document composed by  $N$  sentences), keeping the local coherence estimation feasible on large documents and on large corpora. Formally, the local coherence of a text  $T$  is equal to

$$\begin{aligned} LocalCoherence(T) &= AvgOutDegree(P) \\ &= \frac{1}{N} \sum_{i=1..N} OutDegree(s_i), \end{aligned}$$

where  $OutDegree(s_i)$  is the sum of the weights associated to edges that leave  $s_i$  and  $N$  is the number of sentences in the text. This value can also be seen as the sum of the values of the adjacency matrix of the projection graph (Figures 1(e) and 1(f)) divided by the number of sentences.

## 4 Experiments

We compare our model with the entity grid approach and evaluate the influence of the different weighting schemes used in the projection graphs, either  $P_W$  or  $P_{Acc}$ , where weights are potentially decreased by distance information  $Dist$ . Our baseline corresponds to local coherence computation based on the unweighted projection graph  $P_U$ .

For graph construction, all nouns in a document are considered as discourse entities, even those which do not head NPs as this is beneficial for the entity grid model as described in Elsner and Charniak (2011). We also propose to use a coreference resolution system and consider coreferent entities to be the same discourse entity. To do so, we use one of the top performing systems from the CoNLL 2012 shared task (Martschat et al., 2012). As the coreference resolution system is trained on well-formed textual documents and expects a correct sentence ordering, we use in all our experiments only features that do not rely on sentence order (e.g. alias relations, string matching, etc.).

Grammatical information associated with each entity is extracted automatically thanks to the Stanford parser using dependency conversion (de Marneffe et al., 2006). Syntactic weights in the bipartite graph are defined following the linguistic intuition that subjects are more important than objects, which are themselves more important than other syntactic roles. Preliminary experiments show that as long as weight assignment follows the scheme  $S > O > X$ , then more coherent documents are associated with a higher local coherence value than less coherent document in 90% of cases (while this value equals 49% when no restriction is given on syntactic weights order). Moreover, as the local coherence computation is a linear combination of the syntactic weights, the function is smooth and no large variations of the local coherence values are observed for small changes of weights' values. For these reasons, weights  $w(e, s_i)$  are set as follows: 3 if  $e$  is subject in  $s_i$ , 2 if  $e$  is an object and 1 otherwise.

We evaluate the ability of our graph-based model to estimate the local coherence of a textual document with three different experiments. First, we perform a sentence ordering task (Section 4.1) as proposed in Barzilay and Lapata (2008). Then, as the first task uses "artificial" documents, we also work on two other tasks that involve "real" documents: summary coherence rating (Section 4.2), and readability assessment (Section 4.3). In these experiments, distance computation and syntactic weights are the same for all tasks and all corpora. However, the model is also flexible and can be adapted to the different tasks by optimizing the parameters on a development data set, which may give better results.

### 4.1 Sentence Ordering

The first experiment consists in ranking alternative sentence orderings of a document, as proposed by Barzilay and Lapata (2008) and Elsner and Charniak (2011).

#### 4.1.1 Experimental Settings

The sentence ordering task can be performed in two ways: discrimination and insertion. Discrimination consists in comparing a document to a random permutation of its sentences. For this, our system associates local coherence values with the original document and its permutation, the output of our system being considered as correct if the score for the original document is higher than the

score of its permutation. In the insertion task, proposed by Elsner and Charniak (2011), we evaluate the ability of our system to retrieve the original position of a sentence previously removed from a document. For this, each sentence is removed in turn and a local coherence score is computed for every possible reinsertion position. The system output is considered as correct if the document associated with the highest local coherence score is the one in which the sentence is reinserted in the correct position.

These two tasks were performed on documents extracted from the English test part of the CoNLL 2012 shared task (Pradhan et al., 2012). This corpus, composed by documents of multiple news sources – spoken or written – was preferred to the ACCIDENTS and EARTHQUAKES corpora used by Barzilay and Lapata (2008) for two reasons. First, as mentioned by Elsner and Charniak (2008), these corpora use a very constrained style and are not typical of normal informative documents<sup>3</sup>. Second, we want to evaluate the influence of automatically performed coreference resolution in a controlled fashion. The coreference resolution system used performs well on the CoNLL 2012 data. In this dataset, documents composed by the concatenation of different news articles or too short to have at least 20 permutations were discarded from the corpus. This filtering results in 61 documents composed of 36.1 sentences or 2064 word tokens on average. In both discrimination and insertion, we compare our system against a random baseline where random values are associated with the different orderings.

#### 4.1.2 Discrimination

Accuracy is used to evaluate the ability of our system to discriminate a document from 20 different permutations. It equals the number of times our system gives the highest score to the original document, divided by the number of comparisons. Since the model can give the same score for a permutation and the original document, we also compute F-measure where recall is *correct/total* and precision equals *correct/decisions*. We test significance using the Student’s t-test that can detect significant differences between paired samples. Moreover, as increasing the number of hypotheses

<sup>3</sup>Our graph-based model obtains for the discrimination task an accuracy of 0.846 and 0.635 on the ACCIDENTS and EARTHQUAKES datasets, respectively, compared to 0.904 and 0.872 as reported by Barzilay and Lapata (2008).

	Acc	F	Acc	F
Random	0.496	0.496		
B&L	0.877	0.877		
E&C	0.915	0.915		
	wo coref		w coref	
$P_U, Dist$	0.830	0.830	0.833	0.833
$P_W, Dist$	0.871	0.871	0.849	0.849
$P_{Acc}, Dist$	0.889	0.889	0.852	0.852

Table 3: Discrimination, reproduced baselines (B&L: Barzilay and Lapata (2008); E&C Elsner and Charniak (2011)) vs. graph-based

in a test can also increase the likelihood of witnessing a rare event, and therefore, the chance to reject the null hypothesis when it is true, we use the Bonferroni correction to adjust the increased random likelihood of apparent significance.

Table 3 presents the values obtained by three baseline systems when applied to our corpus. Results for the entity grid models described by Barzilay and Lapata (2008) and Elsner and Charniak (2011) are obtained by using Micha Elsner’s reimplementation in the Brown Coherence Toolkit<sup>4</sup>. The system was trained on the English training part of the CoNLL 2012 shared task filtered in the same way as the test part.

Table 3 also displays the results for our model. These values show that our system performs comparable to the state-of-the-art. Indeed, the difference between our best results and those of Elsner and Charniak are not statistically significant.

In this experiment, distance information is critical. Without it, it is not possible to distinguish between an original document and one of its permutation as both contain the same number and kind of entities. Distance however can detect changes in the distribution of entities within the document as space between entities is significantly modified when sentence order is permuted. When the number of entities “shared” by two sentences is taken into account ( $P_W$ ), the accuracy of our system grows (from 0.830 to 0.871). Table 3 finally shows that syntactic information improves the performance of our system (yet not significantly) and gives the best results ( $P_{Acc}$ ).

We also evaluated the influence of coreference resolution on the performance of our system. Us-

<sup>4</sup><https://bitbucket.org/melsner/browncoherence>; B&L is Elsner’s “baseline entity grid” (command line option ‘-n’), E&C is Elsner’s “extended entity grid” (‘-f’)

	Acc.	Ins.	Acc.	Ins.
Random	0.028	0.071		
E&C	0.068	0.167		
	wo coref		w coref	
$P_U, Dist$	0.062	0.101	0.068	0.120
$P_W, Dist$	0.075	0.114	0.070	0.138
$P_{Acc}, Dist$	0.071	0.102	0.067	0.097

Table 4: Insertion, reproduced baselines vs. graph-based

ing coreference resolution improves the performance of the system when distance information is used alone in the system (Table 3). However, this improvement is not statistically significant.

### 4.1.3 Insertion

Sentence insertion is much more difficult than discrimination for two reasons. First, in insertion, permutations only differ by one sentence. Second, a document is compared to many more permutations in insertion task than in discrimination.

In complement to accuracy, we use the insertion score introduced by Elsner and Charniak (2011) for evaluation. This score – the higher, the better – computes the proximity between the initial and the proposed position of a sentence, averaged by the number of sentences.

Table 4 shows that, as expected, results for this task are much lower than those obtained for discrimination. However they are still comparable with the results of Elsner and Charniak (2011)<sup>5</sup>.

As previously and for the same reasons, distance information is critical for this task. The best results, that present a statistically significant improvement when compared to the random baseline, are obtained when distance information and the number of entities “shared” by two sentences are taken into account ( $P_W$ ). We can see that the accuracy value obtained with our system is higher than the one provided with the entity grid model. However, the entity grid model reaches a significantly higher insertion score. This means that, if it makes more mistakes than our system, the position chosen by the entity grid model is usually closer to the correct position. Finally, contrary to the discrimination task, syntactic information ( $P_{Acc}$ ) does not improve the performance of our system.

<sup>5</sup>Their results are slightly lower than those presented in their paper, probably because our corpus is composed by documents that can be longer than the ones used in their experiments (Wall Street Journal articles).

When the coreference resolution system is used, the best accuracy value decreases while the insertion score increases from 0.114 to 0.138 (Table 4). Therefore, coreference resolution tends to associate positions that are closer to the original ones.

## 4.2 Summary Coherence Rating

To reconfirm the hypothesis that our model can estimate the local coherence of a textual document, we perform a second experiment, summary coherence rating. To this end, we apply our model on the corpus used and proposed by Barzilay and Lapata (2008). As the objective of our model is to estimate the *coherence* of a summary, we prefer this dataset to other summarization evaluation task corpora, as these account for other dimensions of the summaries: content selection, fluency, etc. Starting with a pair of summaries, one slightly more coherent than the other, the objective of the task is to order the two summaries according to local coherence.

### 4.2.1 Experimental Settings

For the summary coherence rating experiment, pairs to be ordered are composed of summaries extracted from the Document Understanding Conference (DUC 2003). Summaries, provided either by humans or by automatic systems, were judged by seven humans annotators and associated with a coherence score (for more details on this score see Barzilay and Lapata (2008)). 80 pairs were then created, each of these being composed by two summaries of a same document where the score of one of the summaries is significantly higher than the score of the second one. Even though all summaries are of approximately the same length (114.2 words on average), their sentence length can vary considerably. Indeed, more coherent summaries tend to have more sentences and contain less entities.

For evaluation purposes, the accuracy still corresponds to the number of correct ratings divided by the number of comparisons, while the F-measure combines recall and precision measures. As before, significance is tested with the Student’s t-test accounting for the Bonferroni correction.

### 4.2.2 Results

Table 5 compares the results reported by Barzilay and Lapata (2008) on the exact same corpus with the results obtained with our system. It shows that

	Acc.	F	Acc.	F
B&L	0.833			
	wo coref		w coref	
$P_U$	0.800	0.815	0.700	0.718
$P_W$	0.613	0.613	0.538	0.548
$P_{Acc}$	0.700	0.704	0.638	0.638
$P_U, Dist$	0.650	0.658	0.550	0.557
$P_W, Dist$	0.525	0.525	0.513	0.513
$P_{Acc}, Dist$	0.700	0.700	0.588	0.588

Table 5: Summary Coherence Rating, reported results from Barzilay and Lapata (2008) vs. graph-based

our system gives results comparable to those obtained by Barzilay and Lapata (2008).

This table also shows that, contrary to sentence ordering task, accounting for the distance between two sentences (*Dist*) tends to decrease the results. This difference is explained by the fact that a manual summary, usually considered as more coherent by humans annotators, tends to contain more (and shorter) sentences than an automatic one. As adding distance information decreases the value of our local coherence score, our graph-based model gives better results without it.

Moreover, in contrast to the first experiment, when accounting for the number of entities “shared” by two sentences ( $P_W$ ), values of accuracy and F-measure are lower. We explain this behaviour by the number of sentences contained in the less coherent documents. Indeed, they are composed by a smaller number of sentences but contain more entities on average. This means that, in these documents, two sentences tend to share a larger number of entities and therefore have a higher local coherence score when the  $P_W$  projection graph is used.

When combined with distance information, syntactic information still improves the results ( $P_{Acc}$ ), though not significantly, but does not lead to the best results for this task.

Finally, Table 5 also shows that using a coreference resolution system for document representation does not improve the performance of our system. We believe that, as mentioned by Barzilay and Lapata (2008), this degradation is related to the fact that automatic summarization systems do not use anaphoric expressions which makes the coreference resolution system useless in this case.

With our graph-based model, the best results are

obtained by the baseline ( $P_U$ ), and experiments show that adding information about distance or syntax does not help in this context. It seems therefore necessary to integrate information that is more appropriate to summaries. Although making the model more appropriate for a specific task is out of the scope of this paper, our model is flexible and accounting for information about genre differences or sentence length, by adding weights in the graph-based representation of the document, is feasible without any modification of the model.

### 4.3 Readability Assessment

Barzilay and Lapata (2008) argue that grid models are domain and style dependent. Therefore they proposed a readability assessment task to test if the entity grid model can be used for style classification. They combined their model with Schwarm and Ostendorf’s (2005) readability features and use Support Vector Machines to classify documents in two categories. With the same intention, we evaluate the ability of our model to differentiate “easy to read” documents from difficult ones.

#### 4.3.1 Experimental Settings

The objective of the readability assessment task is to evaluate how difficult to read a document is. We perform this task on the data used by Barzilay and Lapata (2008), a corpus collected originally by Barzilay and Elhadad (2003) from the *Encyclopedia Britannica* and its version for children, the *Britannica Elementary*. Both versions contain 107 articles. In *Encyclopedia Britannica*, documents are composed by an average of 83.1 sentences while they contain 36.6 sentences in *Britannica Elementary*. Although these texts are not explicitly annotated with grade levels, they represent two broad readability categories.

In order to estimate the complexity of a document, our model computes the local coherence score for each article in the two categories. The article associated with the higher score is considered to be the more readable as it is more coherent, needing less interpretation from the reader than a document associated with a lower local coherence score. Values presented in the following section correspond to accuracy, where the system is correct if it assigns the higher local coherence score to the most “easy to read” document, and F-measure.

	Acc.	F	Acc.	F
S&O	0.786			
B&L	0.509			
B&L + S&O	0.888			
	wo coref		w coref	
$P_U$	0.589	0.589	0.374	0.374
$P_W$	0.579	0.579	0.383	0.383
$P_{Acc}$	0.645	0.645	0.421	0.421
$P_{U, Dist}$	0.589	0.589	0.280	0.280
$P_{W, Dist}$	0.570	0.570	0.290	0.290
$P_{Acc, Dist}$	0.766	0.766	0.308	0.308

Table 6: Readability, reported results from Barzilay and Lapata (2008) vs. graph-based (S&O: Schwarm and Ostendorf (2005))

### 4.3.2 Results

In order to compare our results with those reported by Barzilay and Lapata (2008), entities used for the graph-based representation are discourse entities that head NPs.

Table 6 shows that, for this task, syntactic information plays a dominant role ( $P_{Acc}$ ). A statistically significant improvement is provided by including syntactic information. It gives more weight to subject entities that are more numerous in the *Britannica Elementary* documents which are composed by simpler and shorter sentences. Finally, when distance is accounted for together with syntactic information, the accuracy is significantly improved ( $p < 0.01$ ) with regard to the results obtained with syntactic information only.

Table 6 also shows that when the number of entities “shared” by two sentences is accounted for ( $P_W$ ), the results are lower. Indeed, *Encyclopedia Britannica* documents are composed by longer sentences, that contain a higher number of entities. This increases the local coherence value of difficult documents more than the value of “easy to read” documents, that contain less entities.

When our graph-based representation used the coreference resolution system, unlike the observation of Barzilay and Lapata (2008), the results of our model decrease significantly. The poor performance of our system in this case can be explained by the fact that the coreference resolution system regroups more entities in *Encyclopedia Britannica* documents than in *Britannica Elementary* ones. Therefore, the number of entities that are “shared” by two sentences increases more importantly in the *Encyclopedia Britannica* corpus, while the dis-

tance between two occurrences of one entity decreases in a more significant manner. For these reasons, the coherence scores associated with “difficult to read” documents tend to be higher when coreference resolution is performed on our data, which reduces the performance of our system. As before, syntactic information leads to the best results, but does not allow the accuracy to be higher than random anymore.

Compared to the results provided by Barzilay and Lapata (2008) with the entity grid model alone, our representation outperforms their model significantly. We believe that this difference is caused by how syntactic information is introduced in the document representation and by the fact that our system can deal with entities that appear throughout the whole document while the entity grid model only looks at entities within a three sentences windows. Our model which captures exclusively local coherence is almost on par with the results reported for Schwarm & Ostendorf’s (2005) system which relies on a wide range of lexical, syntactic and semantic features. Only when Barzilay and Lapata (2008) combine the entity grid with Schwarm & Ostendorf’s features they reach performance considerably better than ours.

In addition to the experiments proposed by Barzilay and Lapata (2008), we used a third readability category, the *Britannica Student*, that contains articles targeted for youths (from 11 to 14 years old). These documents, which are quite similar to the *Encyclopedia Britannica* ones, are composed by an average of 44.1 sentences. As we were only able to find 99 articles out of the 107 original ones in this category, sub corpora of the three categories were used for the comparison with the *Britannica Student* articles.

Table 7 shows the results obtained for the comparisons between the two first categories and the *Britannica Student* articles. As previously, coreference resolution tends to lower the results, therefore only values obtained without coreference resolution are reported in the table.

When articles from *Britannica Student* are compared to articles extracted from *Encyclopedia Britannica*, Table 7 shows that the different parameters have the same influence as for comparing between *Encyclopedia Britannica* and *Britannica Elementary*: statistically significant improvement with syntactic information, higher values when distance is taken into account, etc. However, it

	<i>Brit. vs. Stud.</i>		<i>Stud. vs. Elem.</i>	
	Acc.	F	Acc.	F
$P_U$	0.444	0.444	0.667	0.667
$P_W$	0.434	0.434	0.636	0.636
$P_{Acc}$	0.465	0.465	0.707	0.707
$P_U, Dist$	0.475	0.475	0.646	0.646
$P_W, Dist$	0.485	0.485	0.616	0.616
$P_{Acc, Dist}$	0.556	0.556	0.657	0.657

Table 7: Readability, comparison between *Encyclopedia Britannica*, *Britannica Elementary* and *Britannica Student*

can also be seen that accuracy and F-measure are lower for comparing these two corpora. This is probably due to the stylistic difference between these two kinds of articles, which is less significant than the difference between articles from *Encyclopedia Britannica* and *Britannica Elementary*.

Concerning the comparison between *Britannica Student* and *Britannica Elementary* articles, Table 7 shows that integrating distance information gives slightly different results and tends to decrease the values of accuracy and F-measure. This is explained by the fact that *Britannica Elementary* documents contain fewer entities than *Britannica Student* articles. As the length of the two kinds of articles is similar, distance between entities in *Britannica Elementary* documents is more important. As a result, accounting for distance information lowers the local coherence values for the more coherent document, which reduces the performance of our model. As previously, syntactic information improves the results and, for this comparison, the best result is obtained when syntactic information alone is accounted for. This leads to an accuracy which is almost equal to the one when comparing *Encyclopedia Britannica* and *Britannica Elementary* (0.707 against 0.766).

These two additional experiments show that our model is style dependent. It obtains better results when it has to distinguish between *Encyclopedia Britannica* and *Britannica Elementary* or *Britannica Student* and *Britannica Elementary* articles which present a more important difference from a stylistic point of view than articles from *Encyclopedia Britannica* and *Britannica Elementary*.

## 5 Conclusions

In this paper, we proposed an unsupervised and computationally efficient graph-based local coher-

ence model. Experiments show that our model is robust among tasks and domains, and reaches reasonable results for three tasks with the same parameter values and settings (i.e. accuracy values of 0.889, 0.70 and 0.766 for sentence ordering, summary coherence rating and readability assessment tasks respectively ( $P_{Acc}$ ,  $Dist$ )). Moreover, our model can be optimized and obtains results comparable with entity grid based methods when proper settings are used for each task.

Our model has two main advantages over the entity grid model. First, as the graph used for document representation contains information about entity transitions, our model does not need a learning phase. Second, as it relies only on graph centrality, our model does not suffer from the computational complexity and data sparsity problems mentioned by Barzilay and Lapata (2008).

Our current model leaves space for improvement. Future work should first investigate the integration of information about entities. Indeed, our model only uses entities as indications of sentence connection although it has been shown that distinguishing important from unimportant entities, according to their named-entity category, has a positive impact on local coherence computation (Elsner and Charniak, 2011). Moreover, future work should also examine the use of discourse relation information, as proposed in (Lin et al., 2011). This can be easily done by adding edges in the projection graphs when sentences contain entities related from a discourse point of view while Lin et al.’s approach suffers from complexity and data sparsity problems similar to the entity grid model.

Finally, these promising results on local coherence modeling make us believe that our graph-based representation can be used without much modification for other tasks, e.g. extractive summarization or topic segmentation. This could be achieved with link analysis algorithms such as PageRank, that decide on the importance of a (sentence) node within a graph based on global information recursively drawn from the entire graph.

**Acknowledgments.** This work has been funded by the Klaus Tschira Foundation, Heidelberg, Germany. The first author has been supported by a HITS postdoctoral scholarship. We would like to thank Mirella Lapata and Regina Barzilay for making their data available and Micha Elsner for providing his toolkit.

## References

- Regina Barzilay and Noemie Elhadad. 2003. Sentence alignment for monolingual comparable corpora. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Sapporo, Japan, 11–12 July 2003, pages 25–32.
- Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, Mich., 25–30 June 2005, pages 141–148.
- Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.
- Susan E. Brennan, Marilyn W. Friedman, and Carl J. Pollard. 1987. A centering approach to pronouns. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, Stanford, Cal., 6–9 July 1987, pages 155–162.
- Jill Burstein, Joel Tetreault, and Slava Andreyev. 2010. Using entity-based features to model coherence in student essays. In *Proceedings of Human Language Technologies 2010: The Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, Cal., 2–4 June 2010, pages 681–684.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy, 22–28 May 2006, pages 449–454.
- Micha Elsner and Eugene Charniak. 2008. Coreference-inspired coherence modeling. In *Proceedings ACL-HLT 2008 Conference Short Papers*, Columbus, Ohio, 15–20 June 2008, pages 41–44.
- Micha Elsner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *Proceedings of the ACL 2011 Conference Short Papers*, Portland, Oreg., 19–24 June 2011, pages 125–129.
- Micha Elsner, Joseph Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, N.Y., 22–27 April 2007, pages 436–443. Read this version: <http://www.cs.brown.edu/~melsner/order.pdf>.
- Katja Filippova and Michael Strube. 2007. Extending the entity-grid coherence model to semantically related entities. In *Proceedings of the 11th European Workshop on Natural Language Generation*, Schloss Dagstuhl, Germany, 17–20 June 2007, pages 139–142.
- Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.
- Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, 23–26 July 2002, pages 133–142.
- Nikiforos Karamanis, Chris Mellish, Massimo Poesio, and Jon Oberlander. 2009. Evaluating centering for information ordering using corpora. *Computational Linguistics*, 35(1):29–46.
- Rodger Kibble and Richard Power. 2004. Optimizing referential coherence in text generation. *Computational Linguistics*, 30(4):401–416.
- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, Portland, Oreg., 19–24 June 2011, pages 997–1006.
- Sebastian Martschat, Jie Cai, Samuel Broscheit, Éva Mújdricza-Maydt, and Michael Strube. 2012. A multigraph model for coreference resolution. In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning*, Jeju Island, Korea, 12–14 July 2012, pages 100–106.
- Neil McIntyre and Mirella Lapata. 2010. Plot induction and evolutionary search for story generation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 11–16 July 2010, pages 1562–1572.
- Rada Mihalcea and Dragomir Radev. 2011. *Graph-based Natural Language Processing and Information Retrieval*. Cambridge Univ. Press, Cambridge, U.K.
- Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25–55.
- Roberto Navigli and Mirella Lapata. 2010. An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):678–692.
- Mark E.J. Newman. 2010. *Networks: An Introduction*. Oxford University Press, New York, N.Y.
- Massimo Poesio, Rosemary Stevenson, Barbara Di Eugenio, and Janet Hitzeman. 2004. Centering: A parametric theory and its instantiations. *Computational Linguistics*, 30(3): 309–363.
- Sameer Pradhan, Alessandro Moschitti, and Nianwen Xue. 2012. CoNLL-2012 Shared Task: Modeling multilingual unrestricted coreference in OntoNotes.

In *Proceedings of the Shared Task of the 16th Conference on Computational Natural Language Learning*, Jeju Island, Korea, 12–14 July 2012, pages 1–40.

Sarah E. Schwarm and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, Mich., 25–30 June 2005, pages 523–530.

Radu Soricut and Daniel Marcu. 2006. Discourse generation using utility-trained coherence models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, 17–21 July 2006, pages 1105–1112.

# Recognizing Rare Social Phenomena in Conversation: Empowerment Detection in Support Group Chatrooms

Elijah Mayfield, David Adamson, and Carolyn Penstein Rosé

Language Technologies Institute

Carnegie Mellon University

5000 Forbes Ave, Pittsburgh, PA 15213

{emayfiel, dadamson, cprose}@cs.cmu.edu

## Abstract

Automated annotation of social behavior in conversation is necessary for large-scale analysis of real-world conversational data. Important behavioral categories, though, are often sparse and often appear only in specific subsections of a conversation. This makes supervised machine learning difficult, through a combination of noisy features and unbalanced class distributions. We propose *within-instance content selection*, using cue features to selectively suppress sections of text and biasing the remaining representation towards minority classes. We show the effectiveness of this technique in automated annotation of empowerment language in online support group chatrooms. Our technique is significantly more accurate than multiple baselines, especially when prioritizing high precision.

## 1 Introduction

Quantitative social science research has experienced a recent expansion, out of controlled settings and into natural environments. With this influx of interest comes new methodology, and the inevitable question arises of how to move towards testable hypotheses, using these uncontrolled sources of data as scientific lenses into the real world.

The study of conversational transcripts is a key domain in this new frontier. There are certain social and behavioral phenomena in conversation that cannot be easily identified through questionnaire data, self-reported surveys, or easily extracted user metadata. Examples of these social phenomena in conversation include overt displays of power (Prabhakaran et al., 2012) or indicators of rapport and relationship building (Wang et al.,

2012). Manually annotating these social phenomena cannot scale to large data, so researchers turn to automated annotation of transcripts (Rosé et al., 2008). While machine learning is highly effective for annotation tasks with relatively balanced labels, such as sentiment analysis (Pang and Lee, 2004), more complex social functions are often rarer. This leads to unbalanced class label distributions and a much more difficult machine learning task. Moreover, features indicative of rare social annotations tend to be drowned out in favor of features biased towards the majority class. The net effect is that classification algorithms tend to bias towards the majority class, giving low accuracy for rare class detection.

Automated annotation of social phenomena also brings opportunities for real-world applications. For example, real-time annotation of conversation can power adaptive intervention in collaborative learning settings (Rummel et al., 2008; Adamson and Rosé, 2012). However, with the considerable power of automation comes great responsibility. It is critical to avoid intervening in the case of erroneous annotations, as providing unnecessary or inappropriate support in such a setting has been shown to be harmful to group performance and social cohesion (Dillenbourg, 2002; Stahl, 2012).

We propose adaptations to existing machine learning algorithms which improve recognition of rare annotations in conversational text data. Our primary contribution comes in the form of *within-instance content selection*. We develop a novel algorithm based on textual cues, suppressing information which is likely to be irrelevant to an instance's class label. This allows features which predict minority classes to gain prominence, helping to sidestep the frequency of common features pointing to a majority class label.

Additionally, we propose modifications to existing algorithms. First, we identify a new application of logistic model trees to text data. Next,

we define a modification of confidence-based ensemble voting which encourages minority class labeling. Using these techniques, we demonstrate a significant improvement in classifier performance when recognizing the language of *empowerment* in support group chatrooms, a critical application area for researchers studying conversational interactions in healthcare (Uden-Kraan et al., 2009).

The remainder of this paper is structured as follows. We introduce the domain of empowerment in support contexts, along with previous studies on the challenges that these annotations (and similar others) bring to machine learning. We introduce our new technique for improving the ability to automate this annotation, along with other optimizations to the machine learning workflow which are tailored to this skewed class balance. We present experimental results showing that our method is effective, and provide a detailed analysis of the behavior of our model and the features it uses most. We conclude with a discussion of particularly useful applications of this work.

## 2 Background

We ground this paper’s discussion of machine learning with a real problem, turning to the annotation of empowerment language in chat<sup>1</sup>. The concept of empowerment, while a prolific area of research, lacks a broad definition across professionals, but broadly relates to “the power to act efficaciously to bring about desired results” (Boehm and Staples, 2002) and “experiencing personal growth as a result of developing skills and abilities along with a more positive self-definition” (Staples, 1990). Participants in online support groups feel increased empowerment (Uden-Kraan et al., 2009; Barak et al., 2008). Quantitative studies have shown the effect of empowerment through statistical methods such as structural equation modeling (Vauth et al., 2007), as have qualitative methods such as deductive transcript analysis (Owen et al., 2008) and interview studies (Wahlin et al., 2006).

The transition between these styles of research has been gradual. Pioneering work has demonstrated the ability to distinguish empowerment language in written texts, including prompted writing samples (Pennebaker and Seagal, 1999), nar-

<sup>1</sup>Definitions of empowerment are closely related to the notion of *self-efficacy* (Bandura, 1997). For simplicity, we use the former term exclusively in this paper.

Table 1: Empowerment label distribution in our corpus.

Annotation	Label	#	%
Self-Empowerment	NA	1522	79.3
	POS	202	10.5
	NEG	196	10.2
Other-Empowerment	NA	1560	81.3
	POS	217	11.3
	NEG	143	7.4

ratives in online forums (Hoybye et al., 2005), and some preliminary analysis of synchronous discussion (Ogura et al., 2008; Mayfield et al., 2012b). These transitional works have used limited analysis methodology; in the absence of sophisticated natural language processing, their conclusions often rely on coarse measures, such as word counts and proportions of annotations in a text.

Users, of course, do not express empowerment in every thread in which they participate, which leads to a challenge for machine learning. Threads often focus on a single user’s experiences, in which most participants in a chat are merely commentators, if they participate at all, matching previous research on shifts in speaker salience over time (Hassan et al., 2008). This leads to many user threads which are annotated as not applicable (N/A). We move to our proposed approach with these skewed distributions in mind.

## 3 Data

Our data consists of a set of chatroom conversation transcripts from the Cancer Support Community<sup>2</sup>. Each 90-minute conversation took place in the context of a weekly meeting in a real-time chat, with up to 6 participants in addition to a professional therapist facilitating the discussion. In total, 2,206 conversations were collected from 2007-2011. This data offers potentially rich insight into coping and social support; however, annotating such a dataset by hand would be prohibitively expensive, even when it is already transcribed.

Twenty-one of these conversations have been annotated, as originally described and analyzed in (Mayfield et al., 2012b)<sup>3</sup>. This data was disentangled into *threads* based on common themes or topics, as in prior work (Elsner and Charniak,

<sup>2</sup>[www.cancersupportcommunity.org](http://www.cancersupportcommunity.org)

<sup>3</sup>All annotations were found to be adequately reliable between humans, with thread disentanglement  $f = 0.75$  and empowerment annotation  $\kappa > 0.7$ .

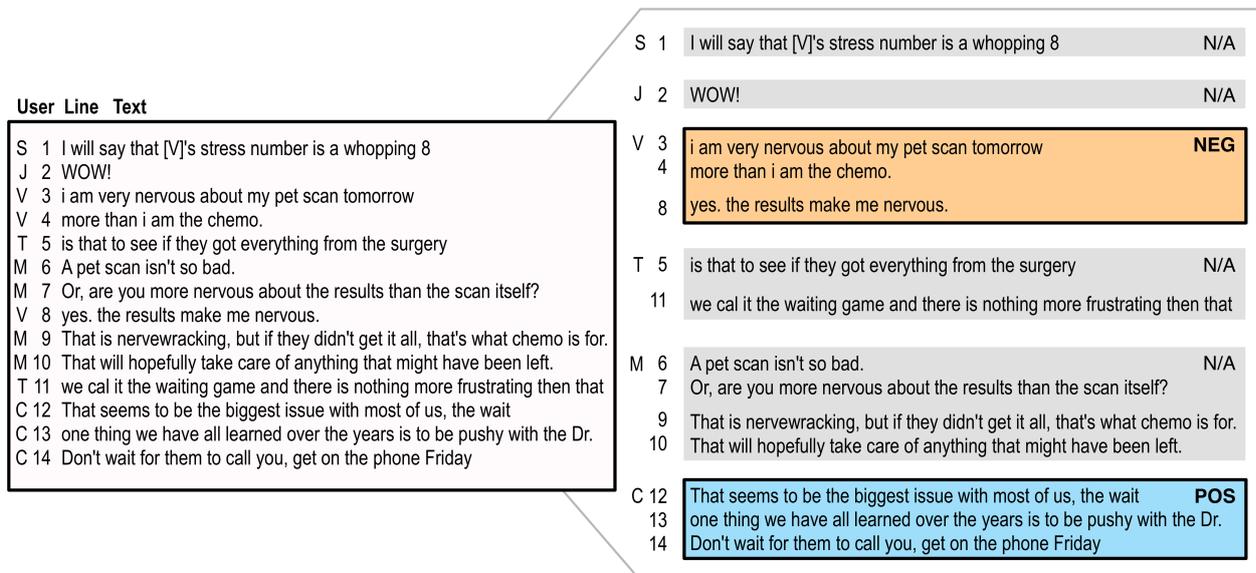


Figure 1: An example mapping from a single thread’s chat lines (left) to the per-user, per-thread instances used for classification in this paper (right), with example annotations for self-empowerment indicated.

2010; Adams and Martel, 2010). A novel *per-user, per-thread* annotation was then employed for empowerment annotation, following a coding manual based on definitions like those in Section 2. Each user was assigned a label of positive or negative empowerment if they exhibited such emotions, or was left blank if they did not do so within the context of that thread. This annotation was performed both for their *self-empowerment* as well as their attitude towards others’ situations (*other-empowerment*). An example of this annotation for self-empowerment is presented in Figure 1 and the distribution of labels is given in Table 1.

Most previous annotation tasks attempt to annotate on a per-utterance basis, such as dialogue act tagging (Popescu-Belis, 2008), or on arbitrary spans of text, such as in the MPQA subjectivity corpus (Wiebe et al., 2005). However, for our task, a per-user, per-thread annotation is more appropriate, because empowerment is often indicated best through narrative (Hoybye et al., 2005). Human annotators are instructed to take this context into account when annotating (Mayfield et al., 2012b). It would therefore be nonsensical to annotate individual lines as “embodying” empowerment. Similar arguments have been made for sentiment, especially as the field moves towards aspect-oriented sentiment (Breck et al., 2007). Assigning labels based on thread boundaries allows for context to be meaningfully taken into account, without crossing topic boundaries.

However, this granularity comes with a price: the distribution of class values in these instances is highly skewed. In our data, the vast majority of users’ threads are marked as not applicable to empowerment. Perhaps more inconveniently, while taking context into account is important for reliable annotation, it leads to extraneous information in many cases. Many threads can have multiple lines of contributions that are topically related to an expression of empowerment (and thus belong in the same thread), but which do not indicate any empowerment themselves. This exacerbates the likelihood of instances being classified as N/A.

We choose to take advantage of these attributes of threads. We know from research in discourse analysis that many sections of conversations are formulaic and rote, like introductions and greetings (Schegloff, 1968). We additionally know that polarity often shifts in dialogue through the use of discourse connectives such as conjunctions and transitional phrases. These issues have been addressed in work in the language technologies community, most notably through the Penn Discourse Treebank (Prasad et al., 2008); however, their applications to noisier synchronous conversation has been rare in computational linguistics.

With these linguistic insights in mind, we examine how we can make best use of them for machine learning performance. While techniques for predicting rare events (Weiss and Hirsh, 1998) and compensating for class imbalance (Frank and

Bouckaert, 2006), these approaches generally focus on statistical properties of large class sets without taking the nature of their datasets into account. In the next section, we propose a new algorithm which takes advantage specifically of the linguistic phenomena in the conversation-based data that we study for empowerment detection. As such, our algorithm is highly suited to this data and task, with the necessary tradeoff in uncertain generality to new domains with unrelated data.

## 4 Cue Discovery for Content Selection

Our algorithm performs content selection by learning a set of cue features. Each of these features indicates some linguistic function within the discourse which should downplay the importance of features either before or after that discourse marker. Our algorithm allows us to evaluate the impact of rules against a baseline, and to iteratively judge each rule atop the changes made by previous rules.

This algorithm fits into existing language technologies research which has attempted to partition documents into sections which are more or less relevant for classification. Many researchers have attempted to make use of cue phrases (Hirschberg and Litman, 1993), especially for segmentation both in prose (Hearst, 1997) and conversation (Galley et al., 2003). The approach of content selection, meanwhile, has been explored for sentiment analysis (Pang and Lee, 2004), where individual sentences may be less subjective and therefore less relevant to the sentiment classification task. It is also similar conceptually to content selection algorithms that have been used for text summarization (Teufel and Moens, 2002) and text generation (Sauper and Barzilay, 2009), both of which rely on finding highly-relevant passages within source texts.

Our work is distinct from these approaches. While we have coarse-grained annotations of empowerment, there is no direct annotation of what makes a good cue for content selection. With our cues, we hope to take advantage of shallow discourse structure in conversation, such as contrastive markers, making use of implicit structure in the conversational domain.

### 4.1 Notation

Before describing extensions to the baseline logistic regression model, we define notation. Our

data is arranged hierarchically. We assume that we have a collection of  $d$  training documents  $\mathbf{Tr} = \{D_1 \dots D_d\}$ , each of which contains many training instances (in our task, an instance consists of all lines of chat from one user in one thread). Our total set of  $n$  instances  $\mathbf{I}$  thus consists of instances  $\{I_1, I_2, \dots I_n\}$ . Each document contains lines of chat  $\mathbf{L}$  and each instance  $I_i$  is comprised of some subset of those lines,  $L_i \subseteq \mathbf{L}$ .

Our feature space  $\mathbf{X} = \{x_1, x_2, \dots x_m\}$  consists of  $m$  unigram features representing the observed vocabulary used in our corpus. Each instance is associated with a feature vector  $\bar{x}$  containing values for each  $x \in \mathbf{X}$ , and each feature  $x$  that is present in the  $i$ -th instance maintains a “memory” of the lines in which it appeared in that instance,  $L_{ix}$ , where  $L_{ix} \subseteq L_i$ . Our potential output labels consist of  $\mathbf{Y} = \{NA, NEG, POS\}$ , though this generalizes to any nominal classification task. Each instance  $I$  is associated with exactly one  $y \in \mathbf{Y}$  for self-empowerment and one for other-empowerment; these two labels do not interact and our tasks are treated as independent in this paper<sup>4</sup>. We define classifiers as functions  $f(\bar{x} \rightarrow y \in \mathbf{Y})$ ; in practice, we use logistic regression via LibLINEAR (Fan et al., 2008).

We define a content selection rule as a pairing  $r = \langle c, t \rangle$  between a cue feature  $c \in \mathbf{X}$  and a selection function  $t \in T$ . We created a list of possible selection functions, given a cue  $c$ , maximizing for generality while being expressive. These are illustrated in Figure 2 and described below:

- **Ignore Local Future (A):** Ignore all features from the two lines after each occurrence of  $c$ .
- **Ignore All Future (B):** Ignore all features occurring after the first occurrence of  $c$ .
- **Ignore Local History (C):** Ignore all features in the two lines preceding each occurrence of  $c$ .
- **Ignore All History (D):** Ignore all features occurring only before the last occurrence of  $c$ .

We define an ensemble member  $E = \langle R, f_R \rangle$  - the ordered list of learned content selection rules  $R = [r_1, r_2, \dots]$  and a classifier  $f_C$  trained on instances transformed by those rules. Our final out-

<sup>4</sup>Future work may examine the interaction of jointly annotating multiple sparse social phenomena.

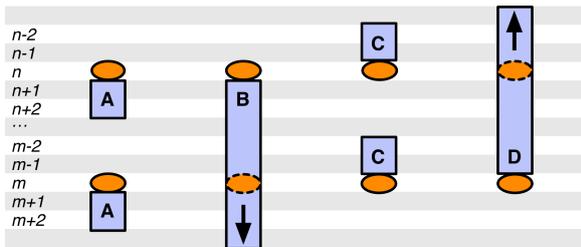


Figure 2: Effects of content selection rules, based on a cue feature (ovals) observed at lines  $m$  and  $n$ .

put of a trained model is a set of ensemble members  $\{E_1, \dots, E_k\}$ .

## 4.2 Algorithm

Our ensemble learning follows the paradigm of cross-validated committees (Parmanto et al., 1996), where  $k$  ensemble members are trained by subdividing our training data into  $k$  subfolds. For each ensemble classifier, cue rules  $R$  are generated on  $k - 1$  subfolds ( $\mathbf{Tr}_k$ ) and evaluated on the remaining subfold ( $\mathbf{Te}_k$ ). In practice, with 21 training documents, 7-fold cross-validation, and  $k = 3$  ensemble members, each generation set consists of 12 documents’ instances, while each evaluation set contains instances from 6 documents.

Our full algorithm is presented in Algorithm 1, and is broken into component parts for clarity. Algorithm 2 begins by measuring the baseline classifier’s ability to recognize minority-class labels. After training on  $\mathbf{Tr}_k$ , we measure the average probability assigned to the correct label of instances in  $\mathbf{Te}_k$ , but only for instances whose correct labels are minority classes (remember, because both  $\mathbf{Tr}_k$  and  $\mathbf{Te}_k$  are drawn from the overall  $\mathbf{Tr}$ , we have access to true class labels). We choose this subset of only minority instances, as we are not interested in optimizing to the majority class.

We next enumerate all rules that we wish to judge. To keep this problem tractable, we ignore features which do not occur in at least 5% of training instances. For the remaining features, we create a candidate rule for each possible pairing of features and selection functions. For each of these candidates, we test its utility by selecting content as if it were an actual rule, then building a new classifier (trained on the generation set) using instances that have been altered in that way. In the evaluation set, we measure the difference in probability of minority class labels being assigned cor-

rectly between the baseline and this altered space. This measure of an individual rule’s impact is described in Algorithm 3.

Once we have evaluated every possible rule once, we select the top-ranked rule and apply it to the feature set. We then iteratively progress through our now-ranked list of candidates, each time treating the newly filtered dataset as our new baseline. We search only top candidates for efficiency, following the fixed-width search methodology for feature selection in very high-dimensionality feature spaces (Gütlein et al., 2009). Each ensemble classifier is finally retrained on all training data, after applying the corresponding content selection rules to that data.

## 5 Prediction

Our prediction algorithm begins with a standard implementation of cross-validated committees (Parmanto et al., 1996), whose results are aggregated with a confidence voting method intended to favor rare labels (Erp et al., 2002). Cross-validated committees are an ensemble technique used to subsample training data to produce multiple hypotheses for classification. Each classifier produced by our cue-based transformation is trained on a subset of our training data. Each makes predictions on all test set instances, producing a distribution of confidence across possible labels. These values serve as inputs to a voting method to produce a final label for each instance.

Compared to other ensemble methods, cross-validated committees as described above are a good fit for our task, because of its unique unit of analysis. As thread-level analysis is the set of individual participants’ turns in a conversation, we risk overfitting if we sample from the same conversations for the training and testing sets. In contrast to standard bagging, hard sampling boundaries never train and test on instances drawn from the same conversation.

To aggregate the votes from members of this ensemble into a final prediction, we employ a variant on Selfridge’s Pandemonium (Selfridge, 1958). If a minority label is selected as the highest-confidence value in any classifier in our ensemble, it is selected. The majority label, by contrast, is only selected if it is the most likely prediction by all classifiers in our ensemble. Thus consensus is required to elect the majority class, and the strongest minority candidate is elected otherwise.

**In** : generation set  $\mathbf{Tr}_k$ , evaluation set  $\mathbf{Te}_k$   
**Out**: ensemble committee  $\{E_1 \dots E_k\}$   
**for**  $i = 1$  **to**  $k$  **do**  
     $R_{final} \leftarrow []$ ;  
     $\mathbf{X}_{freq} \leftarrow \{x \in \mathbf{X} \mid freq(x) \in \mathbf{Tr}_k > 5\%\}$ ;  
     $R \leftarrow \mathbf{X}_{freq} \times T$ ;  
     $R^* \leftarrow R$ ;  
    **repeat**  
         $P_{base} \leftarrow \text{EvaluateClassifier}(\mathbf{Tr}_k, \mathbf{Te}_k)$ ;  
         $\text{EvaluateRules}(P_{base}, \mathbf{Tr}_k, \mathbf{Te}_k, R^*)$ ;  
         $\mathbf{Tr}_k, \mathbf{Te}_k \leftarrow \text{ApplyRule}(R^*[0])$ ;  
         $R \leftarrow R - R^*[0]$ ;  
         $\Delta \leftarrow score(R^*[0])$ ;  
         $R_{final} \leftarrow R_{final} + R^*[0]$ ;  
         $R^* \leftarrow R[0 \dots 50]$ ;  
    **until**  $\Delta < threshold$ ;  
     $\mathbf{Tr}_{final} \leftarrow \mathbf{Tr}_k \cup \mathbf{Te}_k$ ;  
    **foreach**  $r \in R_{final}$  **do**  
         $\mathbf{Tr}_{final} \leftarrow \text{ApplyRule}(\mathbf{Tr}_{final}, r)$ ;  
    **end**  
    Train  $f(\bar{x} \rightarrow y)$  on  $\mathbf{Tr}_{final}$ ;  
**end**

**Algorithm 1:** LearnSelectionCues()

This approach is designed to bias the prediction of our machine learning algorithms in favor of minority classes in a coherent manner. If there is a plausible model that has been trained which recognizes the possibility of a rare label, it is used; the prediction only reverts to the majority class when no plausible minority label could be chosen. As validation of this technique, we compare our “minority pandemonium” approach against both typical pandemonium and standard sum-rule confidence voting (Erp et al., 2002).

### 5.1 Logistic Model Stumps

One characteristic of highly skewed data is that, while minority labels may be expressed in a number of different surface forms, there are many obvious cases in which they do not apply. These cases can actually be harmful to classification of borderline cases. Features that could be given high weight in marginal cases may be undervalued in “low-hanging fruit” easy cases. To remove those obvious instances, a very simple screening heuristic is often enough to eliminate frequent phenotypes of instances where the rare annotation is not present. Prior work has sometimes screened training data through obvious heuristic rules, espe-

**In** : generation set  $\mathbf{Tr}_k$ , evaluation set  $\mathbf{Te}_k$   
**Out**: minority class probability average  $P_{base}$   
Train  $f(\bar{x} \rightarrow y)$  on  $\mathbf{Tr}_k$ ;  
 $\mathbf{Te}_k^{min} \leftarrow \{Instance I \in \mathbf{Te}_k \mid y_I \neq \text{“NA”}\}$   
;  
 $P_{base} \leftarrow 0$ ;  
**foreach**  $Instance I \in \mathbf{Te}_k^{min}$  **do**  
     $P_{base} \leftarrow P_{base} + P(f(\bar{x}_I) = y_I)$   
**end**  
 $P_{base} = P_{base} / size(\mathbf{Te}_k^{min})$   
**Algorithm 2:** EvaluateClassifier()  
  
**In** :  $\mathbf{Tr}_k, \mathbf{Te}_k$ , rules  $R$ , base probability  $P_{base}$   
**Out**:  $R$  sorted on each rule’s improvement score  
**foreach**  $Rule r \in R$  **do**  
     $\mathbf{Tr}'_k, \mathbf{Te}'_k \leftarrow \text{ApplyRule}(\mathbf{Tr}_k, \mathbf{Te}_k, r)$ ;  
     $P_{alter} \leftarrow \text{EvaluateClassifier}(\mathbf{Tr}'_k, \mathbf{Te}'_k)$ ;  
     $score(r) \leftarrow P_{alter} - P_{base}$ ;  
**end**  
Sort  $R$  on  $score(r)$  from high to low;  
**Algorithm 3:** EvaluateRules()

cially in speech recognition; for instance, training speech recognition for words followed by a pause separately from words followed by another word (Franco et al., 2010), or training separate models based on gender (Jiang et al., 1999).

We achieve this instance screening by learning logistic model tree stumps (Landwehr et al., 2005), which allow us to quickly partition data if there is a particularly easy heuristic that can be learned to eliminate a large number of majority-class labels. One challenge of this approach is our underlying unigram feature space - tree-based algorithms are generally poor classifiers for the high-dimensionality, low-information features in a lexical feature space (Han et al., 2001). To compensate, we employ a smaller, denser set of binary features for tree stump screening: instance length thresholds and LIWC category membership.

First, we define a set of features that split based on the number of lines an instance contains, from 1 to 10 (only a tiny fraction of instances are more than 10 lines long). For example, a feature splitting on instances with lines  $\leq 2$  would be true for one- and two-line instances, and false for all others. Second, we define a feature for each category in the Linguistic Inquiry and Word Count dictionary (Tausczik and Pennebaker, 2010) - these broad classes of words allow for more balanced

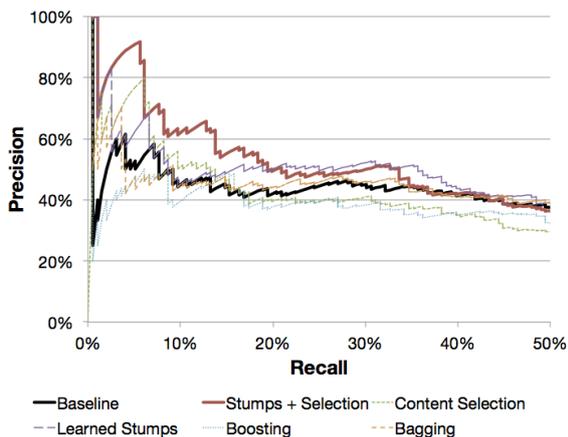


Figure 3: Precision/recall curves for algorithms. After 50% recall all models converge and there are no significant differences in performance.

splits than would unigrams alone. Each category’s feature is true if any word in that category was used at least once in that instance.

We exhaustively sweep this feature space, and report the most successful stump rules for each annotation task. In our other experiments, we report results with and without the best rule for this preprocessing step; we also measure its impact alone.

## 6 Experimental Results

All experiments were performed using LightSIDE (Mayfield and Rosé, 2013). We use a binary unigram feature space, and we perform 7-fold cross-validation. Instances from the same chat transcript never occur in both train and testing folds. Furthermore, we assume that threads have been disentangled already, and our experiments use gold standard thread structure. While this is not a trivial assumption, prior work has shown thread disentanglement to be manageable (Mayfield et al., 2012a); we consider it an acceptable simplifying assumption for our experiments. We compare our methods against baselines including a majority baseline, a baseline logistic regression classifier with L2 regularized features, and two common ensemble methods, AdaBoost (Freund and Schapire, 1996) and bagging (Breiman, 1996) with logistic regression base classifiers<sup>5</sup>.

Table 2 presents the best-performing result from each classification method. For self-empowerment recognition, all methods that we introduce are significant improvements in  $\kappa$ , the

<sup>5</sup>These methods usually use weak, unstable base classifiers; however, in our experiments, those performed poorly.

Table 2: Performance for baselines, common ensemble algorithms, and proposed methods. Statistically significant improvements over baseline are marked ( $p < .01$ , †;  $p < .05$ , \*;  $p < 0.1$ , +).

Method	Self		Other	
	%	$\kappa$	%	$\kappa$
Majority	79.3	.000	81.3	.000
LR Baseline	81.0	.367	81.0	.270
LR + Boosting	78.1	.325	78.5	.275
LR + Bagging	81.2	.352	<b>81.9</b>	.265
LR + Committee	81.0	.367	81.0	.270
Learned Stumps	<b>81.8*</b>	.385†	81.7	<b>.293+</b>
Content Selection	80.9	.389†	80.7	.282
Stumps+Selection	81.3	<b>.406†</b>	79.4	.254

Table 3: Performance of content-selection wrapped learners, for minority voting and two baseline voting methods.

Method	Self		Other	
	%	$\kappa$	%	$\kappa$
Pandemonium	80.3	.283	81.4	.239
Averaged	80.6	.304	<b>81.6</b>	.251
Minority Voting	<b>80.9†</b>	<b>.389†</b>	80.7	<b>.282</b>

measurement of agreement over chance, compared to all baselines. While accuracy remains stable, this is due to predictions shifting away from the majority class and towards minority classes. Our combined model using both logistic model tree stumps and content selection is significantly better than either alone ( $p < .01$ ). To compare the minority pandemonium voting method against baselines of simple pandemonium and summed confidence voting, Table 3 presents the results of content selection wrappers with each voting method. Minority voting is more effective compared to standard confidence voting, improving  $\kappa$  while modestly reducing accuracy; this is typical of a shift towards minority class predictions.

## 7 Discussion

These results show promise for our techniques, which are able to distinguish features of rare labels, previously awash in a sea of irrelevance. Figure 3 shows the impact of our rules as we tune to different levels of recall, with a large boost in precision when recall is not important; our model converges with the baseline for high-recall, low-precision tuning. This suggests that our method is particularly suitable for tasks where confident la-

Table 4: Cue rules commonly selected by the algorithm. Average improvement over the LR baseline is also shown.

<b>Self-Empowerment</b>		
Cue	Transformation	$\Delta\%$
<i>and,but</i>	Ignore Local Future	+5.0
<i>have</i>	Ignore All History	+4.3
<i>!</i>	Ignore All History	+4.2
<i>me,my</i>	Ignore All History	+3.4
<b>Other-Empowerment</b>		
Cue	Transformation	$\Delta\%$
<i>and,but</i>	Ignore Local Future	+5.5
<i>you</i>	Ignore Local History	+5.2
<i>'s</i>	Ignore Local History	+4.1
<i>that</i>	Ignore Local History	+3.9

being of a few instances is more important than labeling as many instances as possible. This is common when tasks have a high cost or carry high risk (for instance, providing real-time conversational supports with an agent, where inappropriate intervention could be disruptive). Other low-recall applications include exploration large corpora for exemplar instances, where the most confident predictions for a given label should be presented first for analyst use. In the rest of this section, we examine notable within-instance and per-instance rules selected by our methods. These rules are summarized in Tables 4 and 5.

For both self- and other-empowerment, we find pronoun rules that match the task (first-person and second-person pronouns for self-Empowerment and other-Empowerment respectively). In both tasks, we find cue rules that suppress the context preceding personal pronouns. These, as well as the possessive suffix *'s*, echo the per-instance effect of the *Self* and *You* splits, anticipating that what follows such a personal reference is likely to bear an evaluation of empowerment. Exclamation marks may indicate strong emotion - we find many instances where what precedes a line with an exclamation is more objective, and what follows includes an assessment. Conjunctions *but* and *and* are selected as cue rules suppressing the two lines that follow the occurrence - suggesting, as suspected, that connective discourse markers play a role in indicating empowerment (Fraser, 1999).

The best-performing stump splits for the Self-Empowerment annotation are *Line Length*  $\leq 1$  and the LIWC word-categories *Article*, *Swear*, and

Table 5: Best decision rules for logistic model stumps. Significant improvement ( $p < 0.05$ ) indicated with \*.

<b>Self-Empowerment</b>				
Split Rule	$\kappa$	$\Delta\kappa$	%	$\Delta\%$
Split $\leq 1$ *	0.385	+0.018	81.8	+0.8
LIWC-Article	0.379	+0.012	81.6	+0.6
LIWC-Swear *	0.376	+0.009	81.4	+0.4
LIWC-Self *	0.376	+0.009	81.5	+0.5
<b>Other-Empowerment</b>				
Split Rule	$\kappa$	$\Delta\kappa$	%	$\Delta\%$
LIWC-You	0.293	+0.023	81.7	+0.7
LIWC-Eating *	0.283	+0.013	81.6	+0.6
LIWC-Negate *	0.282	+0.012	82.3	+1.3
LIWC-Present	0.281	+0.011	81.6	+0.6

*Self*. The split on line length corresponds to the observation that longer instances provide greater opportunity for personal narrative self-assessment to occur (95% of single-line instances are labeled NA). The *Article* category may serve as a proxy for content length - article-less instances in our corpus include one-line social greetings and exchanges of contact information. *Swear* words may be a cue for awareness of self-empowerment - a recent study of women coping with illness reported that swearing in the presence of others, but not alone, was related to potentially harmful outcomes (Robbins et al., 2011). Among *other-* oriented split rules, *Eating* stands out as non-obvious, although medical literature has suggested a link between dietary behavior and empowerment attitudes in a study of women with cancer (Pinto et al., 2002).

## 8 Conclusion

We have demonstrated an algorithm for improving automated classification accuracy on highly skewed tasks for conversational data. This algorithm, particularly its focus on content selection, is rooted in the structural format of our data, which can generalize to many tasks involving conversational data. Our experiments show that this model significantly improves machine learning performance. Our algorithm is taking advantage of structural facets of discourse markers, lending basic sociolinguistic validity to its behavior. Though we have treated each of these rarely-occurring labels as independent thus far, in practice we know that this is not the case. Joint prediction of labels through structured modeling is an obvious next

step for improving classification accuracy.

This is an important step towards large-scale analysis of the impact of support groups on patients and caregivers. Our method can be used to confidently highlight occurrences of rare labels in large data sets. This has real-world implications for professional intervention in social conversational domains, especially in scenarios where such an intervention is likely to be associated with a high cost or high risk. With the construction of more accurate classifiers, we open the possibility of automating annotation on large conversational datasets, enabling new directions for researchers with domain expertise.

## Acknowledgments

The research reported here was supported by National Science Foundation grant IIS-0968485.

## References

- Paige Adams and Craig Martel. 2010. Conversational thread extraction and topic detection in text-based chat. In *Semantic Computing*.
- David Adamson and Carolyn Penstein Rosé. 2012. Coordinating multi-dimensional support in collaborative conversational agents. In *Proceedings of Intelligent Tutoring Systems*.
- Albert Bandura. 1997. *Self-Efficacy: The Exercise of Control*.
- Azy Barak, Meyran Boniel-Nissim, and John Suler. 2008. Fostering empowerment in online support groups. *Computers in Human Behavior*.
- A Boehm and L H Staples. 2002. The functions of the social worker in empowering: The voices of consumers and professionals. *Social Work*.
- Eric Breck, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of IJCAI*.
- Leo Breiman. 1996. Bagging predictors. *Machine Learning*.
- Pierre Dillenbourg. 2002. Over-scripting cscl: The risks of blending collaborative learning with instructional design. *Three worlds of CSCL. Can we support CSCL?*
- Micha Elsner and Eugene Charniak. 2010. Disentangling chat. *Computational Linguistics*.
- Merijn Van Erp, Louis Vuurpijl, and Lambert Schomaker. 2002. An overview and comparison of voting methods for pattern recognition. In *Frontiers in Handwriting Recognition*. IEEE.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification.
- Horacio Franco, Harry Bratt, Romain Rossier, Venkata Rao Gadde, Elizabeth Shriberg, Victor Abrash, and Kristin Precoda. 2010. Eduspeak: A speech recognition and pronunciation scoring toolkit for computer-aided language learning applications. *Language Testing*.
- Eibe Frank and Remco R Bouckaert. 2006. Naive bayes for text classification with unbalanced classes. *Knowledge Discovery in Databases*.
- Bruce Fraser. 1999. What are discourse markers? *Journal of pragmatics*, 31(7):931–952.
- Yoav Freund and Robert E Schapire. 1996. Experiments with a new boosting algorithm. In *Proceedings of ICML*.
- Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of ACL*.
- Martin Gütlein, Eibe Frank, Mark Hall, and Andreas Karwath. 2009. Large-scale attribute selection using wrappers. In *Proceedings of IEEE CIDM*.
- Eui-Hong Han, George Karypis, and Vipin Kumar. 2001. Text categorization using weight adjusted k-nearest neighbor classification. *Lecture Notes in Computer Science: Advances in Knowledge Discovery and Data Mining*.
- Ahmed Hassan, Anthony Fader, Michael H Crespin, Kevin M Quinn, Burt L Monroe, Michael Colaresi, and Dragomir R Radev. 2008. Tracking the dynamic evolution of participant salience in a discussion. In *Proceedings of Coling*.
- Marti A Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*.
- Julia Hirschberg and Diane Litman. 1993. Empirical studies on the disambiguation of cue phrases. *Computational Linguistics*.
- Mette Terp Hoybye, Christoffer Johansen, and Tine Tjørnhøj-Thomsen. 2005. Online interaction effects of storytelling in an internet breast cancer support group. *Psycho-oncology*.
- Hui Jiang, Keikichi Hirose, and Qiang Huo. 1999. Robust speech recognition based on a bayesian prediction approach. In *IEEE Transactions on Speech and Audio Processing*.
- Niels Landwehr, Mark Hall, and Eibe Frank. 2005. Logistic model trees. *Machine Learning*.
- Elijah Mayfield and Carolyn Penstein Rosé. 2013. Lightside: Open source machine learning for text. In *Handbook of Automated Essay Evaluation: Current Applications and New Directions*.

- Elijah Mayfield, David Adamson, and Carolyn Penstein Rosé. 2012a. Hierarchical conversation structure prediction in multi-party chat. In *Proceedings of SIGDIAL Meeting on Discourse and Dialogue*.
- Elijah Mayfield, Miaomiao Wen, Mitch Golant, and Carolyn Penstein Rosé. 2012b. Discovering habits of effective online support group chatrooms. In *ACM Conference on Supporting Group Work*.
- Kanayo Ogura, Takashi Kusumi, and Asako Miura. 2008. Analysis of community development using chat logs: A virtual support group of cancer patients. In *Proceedings of the IEEE Symposium on Universal Communication*.
- Jason E. Owen, Erin O’Carroll Bantum, and Mitch Golant. 2008. Benefits and challenges experienced by professional facilitators of online support groups for cancer survivors. In *Psycho-Oncology*.
- Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the Association for Computational Linguistics*.
- Bambang Parmanto, Paul Munro, and Howard R Doyle. 1996. Improving committee diagnosis with resampling techniques. In *Proceedings of NIPS*.
- James W Pennebaker and J D Seagal. 1999. Forming a story: The health benefits of narrative. *Journal of Clinical Psychology*.
- Bernardine M Pinto, Nancy C Maruyama, Matthew M Clark, Dean G Cruess, Elyse Park, and Mary Roberts. 2002. Motivation to modify lifestyle risk behaviors in women treated for breast cancer. In *Mayo Clinic Proceedings*.
- Andrei Popescu-Belis. 2008. Dimensionality of dialogue act tagsets: An empirical analysis of large corpora. In *Language Resources and Evaluation*.
- Vinodkumar Prabhakaran, Owen Rambow, and Mona Diab. 2012. Predicting overt display of power in written dialogs. In *Proceedings of NAACL*.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Milt-sakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of LREC*.
- Megan L Robbins, Elizabeth S Focella, Shelley Kasle, Ana María López, Karen L Weihs, and Matthias R Mehl. 2011. Naturalistically observed swearing, emotional support, and depressive symptoms in women coping with illness. *Health Psychology*, 30:789.
- Carolyn Penstein Rosé, Yi-Chia Wang, Yue Cui, Jaime Arguello, Karsten Stegmann, Armin Weinberger, and Frank Fischer. 2008. Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning. In *International Journal of Computer Supported Collaborative Learning*.
- Nikol Rummel, Armin Weinberger, Christof Wecker, Frank Fischer, Anne Meier, Eleni Voyiatzaki, George Kahrmanis, Hans Spada, Nikolaos Avouris, and Erin Walker. 2008. New challenges in cscl: Towards adaptive script support. In *Proceedings of ICLS*.
- Christina Sauper and Regina Barzilay. 2009. Automatically generating wikipedia articles: A structure-aware approach. In *Proceedings of ACL*.
- Emanuel A Schegloff. 1968. Sequencing in conversational openings. *American Anthropologist*.
- Oliver G Selfridge. 1958. Pandemonium: a paradigm for learning. In *Proceedings of Symposium on Mechanisation of Thought Processes, National Physical Laboratory*.
- Gerry Stahl. 2012. Interaction analysis of a biology chat. *Productive multivocality*.
- Lee H Staples. 1990. Powerful ideas about empowerment. *Administration in Social Work*.
- Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of Language and Social Psychology*.
- Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics*.
- C F Van Uden-Kraan, C H C Drossaert, E Taal, E R Seydel, and M A F J Van de Laar. 2009. Participation in online patient support groups endorses patients empowerment. *Patient Education and Counseling*.
- R Vauth, B Kleim, M Wirtz, and P W Corrigan. 2007. Self-efficacy and empowerment as outcomes of self-stigmatizing and coping in schizophrenia. *Psychiatry Research*.
- Ingrid Wahlin, Anna-Christina Ek, and Ewa Idvali. 2006. Patient empowerment in intensive carean interview study. *Intensive and Critical Care Nursing*.
- William Yang Wang, Samantha Finkelstein, Amy Ogan, Alan Black, and Justine Cassell. 2012. “love ya, jerkface:” using sparse log-linear models to build positive (and impolite) relationships with teens. In *Proceedings of SIGDIAL*.
- Gary M Weiss and Haym Hirsh. 1998. Learning to predict rare events in event sequences. In *Proceedings of KDD*.
- Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*.

# Decentralized Entity-Level Modeling for Coreference Resolution

Greg Durrett, David Hall, and Dan Klein

Computer Science Division

University of California, Berkeley

{gdurrett, dlwh, klein}@cs.berkeley.edu

## Abstract

Efficiently incorporating entity-level information is a challenge for coreference resolution systems due to the difficulty of exact inference over partitions. We describe an end-to-end discriminative probabilistic model for coreference that, along with standard pairwise features, enforces structural agreement constraints between specified properties of coreferent mentions. This model can be represented as a factor graph for each document that admits efficient inference via belief propagation. We show that our method can use entity-level information to outperform a basic pairwise system.

## 1 Introduction

The inclusion of entity-level features has been a driving force behind the development of many coreference resolution systems (Luo et al., 2004; Rahman and Ng, 2009; Haghighi and Klein, 2010; Lee et al., 2011). There is no polynomial-time dynamic program for inference in a model with arbitrary entity-level features, so systems that use such features typically rely on making decisions in a pipelined manner and sticking with them, operating greedily in a left-to-right fashion (Rahman and Ng, 2009) or in a multi-pass, sieve-like manner (Raghunathan et al., 2010). However, such systems may be locked into bad coreference decisions and are difficult to directly optimize for standard evaluation metrics.

In this work, we present a new structured model of entity-level information designed to allow efficient inference. We use a log-linear model that can be expressed as a factor graph. Pairwise features appear in the model as unary factors, adjacent to nodes representing a choice of antecedent (or none) for each mention. Additional nodes model entity-level properties on a per-mention basis, and

structural agreement factors softly drive properties of coreferent mentions to agree with one another. This is a key feature of our model: mentions manage their partial membership in various coreference chains, so that information about entity-level properties is decentralized and propagated across individual mentions, and we never need to explicitly instantiate entities.

Exact inference in this factor graph is intractable, but efficient approximate inference can be carried out with belief propagation. Our model is the first discriminatively-trained model that both makes joint decisions over an entire document and models specific entity-level properties, rather than simply enforcing transitivity of pairwise decisions (Finkel and Manning, 2008; Song et al., 2012).

We evaluate our system on the dataset from the CoNLL 2011 shared task using three different types of properties: synthetic oracle properties, entity phi features (number, gender, animacy, and NER type), and properties derived from unsupervised clusters targeting semantic type information. In all cases, our transitive model of entity properties equals or outperforms our pairwise system and our reimplementation of a previous entity-level system (Rahman and Ng, 2009). Our final system is competitive with the winner of the CoNLL 2011 shared task (Lee et al., 2011).

## 2 Example

We begin with an example motivating our use of entity-level features. Consider the following excerpt concerning two famous auction houses:

*When looking for [art items], [people] go to [Sotheby's and Christie's] because [they]<sub>A</sub> believe [they]<sub>B</sub> can get the best price for [them].*

The first three mentions are all distinct entities, *they<sub>A</sub>* and *they<sub>B</sub>* refer to *people*, and *them* refers to *art items*. The three pronouns are tricky to resolve

automatically because they could at first glance resolve to any of the preceding mentions. We focus in particular on the resolution of *they*<sub>A</sub> and *them*. In order to correctly resolve *they*<sub>A</sub> to *people* rather than *Sotheby's and Christie's*, we must take advantage of the fact that *they*<sub>A</sub> appears as the subject of the verb *believe*, which is much more likely to be attributed to people than to auction houses.

Binding principles prevent *them* from attaching to *they*<sub>B</sub>. But how do we prevent it from choosing as its antecedent the next closest agreeing pronoun, *they*<sub>A</sub>? One way is to exploit the correct coreference decision we have already made, *they*<sub>A</sub> referring to *people*, since people are not as likely to have a price as art items are. This observation argues for enforcing agreement of entity-level semantic properties during inference, specifically properties relating to permitted semantic roles. Because even these six mentions have hundreds of potential partitions into coreference chains, we cannot search over partitions exhaustively, and therefore we must design our model to be able to use this information while still admitting an efficient inference scheme.

### 3 Models

We will first present our BASIC model (Section 3.1) and describe the features it incorporates (Section 3.2), then explain how to extend it to use transitive features (Sections 3.3 and 3.4).

Throughout this section, let  $x$  be a variable containing the words in a document along with any relevant precomputed annotation (such as parse information, semantic roles, etc.), and let  $n$  denote the number of mentions in a given document.

#### 3.1 BASIC Model

Our BASIC model is depicted in Figure 1 in standard factor graph notation. Each mention  $i$  has an associated random variable  $a_i$  taking values in the set  $\{1, \dots, i-1, \langle \text{new} \rangle\}$ ; this variable specifies mention  $i$ 's selected antecedent or indicates that it begins a new coreference chain. Let  $a = (a_1, \dots, a_n)$  be the vector of the  $a_i$ . Note that a set of coreference chains  $C$  (the final desired output) can be uniquely determined from  $a$ , but  $a$  is not uniquely determined by  $C$ .

We use a log linear model of the conditional distribution  $P(a|x)$  as follows:

$$P(a|x) \propto \exp \left( \sum_{i=1}^n \mathbf{w}^T \mathbf{f}_A(i, a_i, x) \right)$$

When looking for [art items], [people] go to [Sotheby's and Christie's] because [they]<sub>A</sub> believe [they]<sub>B</sub> can get the best price for [them].

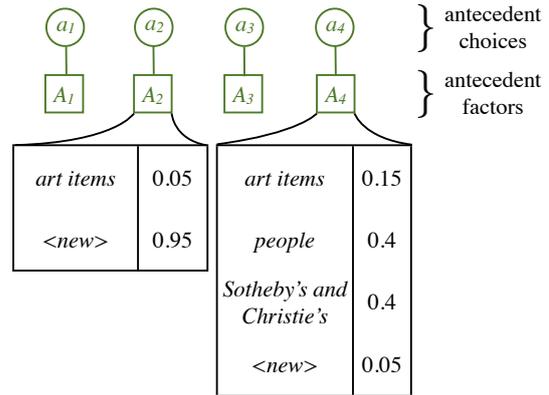


Figure 1: Our BASIC coreference model. A decision  $a_i$  is made independently for each mention about what its antecedent mention should be or whether it should start a new coreference chain. Each unary factor  $A_i$  has a log-linear form with features examining mention  $i$ , its selected antecedent  $a_i$ , and the document context  $x$ .

where  $\mathbf{f}_A(i, a_i, x)$  is a feature function that examines the coreference decision  $a_i$  for mention  $i$  with document context  $x$ ; note that this feature function can include pairwise features based on mention  $i$  and the chosen antecedent  $a_i$ , since information about each mention is contained in  $x$ .

Because the model factors completely over the individual  $a_i$ , these feature functions  $\mathbf{f}_A$  can be expressed as unary factors  $A_i$  (see Figure 1), with  $A_i(j) \propto \exp(\mathbf{w}^T \mathbf{f}_A(i, j, x))$ . Given a setting of  $\mathbf{w}$ , we can determine  $\hat{a} = \arg \max_a P(a|x)$  and then deterministically compute  $C(\hat{a})$ , the final set of coreference chains.

While the features of this model factor over coreference links, this approach differs from classical pairwise systems such as Bengtson and Roth (2008) or Stoyanov et al. (2010). Because potential antecedents compete with each other and with the non-anaphoric hypothesis, the choice of  $a_i$  actually represents a joint decision about  $i-1$  pairwise links, as opposed to systems that use a pairwise binary classifier and a separate agglomeration step, which consider one link at a time during learning. This approach is similar to the mention-ranking model of Rahman and Ng (2009).

#### 3.2 Pairwise Features

We now present the set of features  $\mathbf{f}_A$  used by our unary factors  $A_i$ . Each feature examines the an-

tecedent choice  $a_i$  of the current mention as well as the observed information  $x$  in the document. For each of the features we present, two conjoined versions are included: one with an indicator of the type of the current mention being resolved, and one with an indicator of the types of the current and antecedent mentions. Mention types are either NOMINAL, PROPER, or, if the mention is pronominal, a canonicalized version of the pronoun abstracting away case.<sup>1</sup>

Several features, especially those based on the precise constructs (apposition, etc.) and those incorporating phi feature information, are computed using the machinery in Lee et al. (2011). Other features were inspired by Song et al. (2012) and Rahman and Ng (2009).

**Anaphoricity features:** Indicator of anaphoricity, indicator on definiteness.

**Configurational features:** Indicator on distance in mentions (capped at 10), indicator on distance in sentences (capped at 10), does the antecedent c-command the current mention, are the two mentions in a subject/object construction, are the mentions nested, are the mentions in deterministic appositive/role appositive/predicate nominative/relative pronoun constructions.

**Match features:** Is one mention an acronym of the other, head match, head contained (each way), string match, string contained (each way), relaxed head match features from Lee et al. (2011).

**Agreement features:** Gender, number, animacy, and NER type of the current mention and the antecedent (separately and conjoined).

**Discourse features:** Speaker match conjoined with an indicator of whether the document is an article or conversation.

Because we use conjunctions of these base features together with the antecedent and mention type, our system can capture many relationships that previous systems hand-coded, especially regarding pronouns. For example, our system has access to features such as “*it* is non-anaphoric”, “*it* has as its antecedent a geopolitical entity”, or “*I* has as its antecedent *I* with the same speaker.”

<sup>1</sup>While this canonicalization could theoretically impair our ability to resolve, for example, reflexive pronouns, conjoining features with raw pronoun strings does not improve performance.

We experimented with synonymy and hypernymy features from WordNet (Miller, 1995), but these did not empirically improve performance.

### 3.3 TRANSITIVE Model

The BASIC model can capture many relationships between pairs of mentions, but cannot necessarily capture entity-level properties like those discussed in Section 2. We could of course model entities directly (Luo et al., 2004; Rahman and Ng, 2009), saying that each mention refers to some prior entity rather than to some prior mention. However, inference in this model would require reasoning about all possible partitions of mentions, which is computationally infeasible without resorting to severe approximations like a left-to-right inference method (Rahman and Ng, 2009).

Instead, we would like to try to preserve the tractability of the BASIC model while still being able to exploit entity-level information. To do so, we will allow *each mention* to maintain its own distributions over values for a number of properties; these properties could include gender, named-entity type, or semantic class. Then, we will require each anaphoric mention to agree with its antecedent on the value of each of these properties.

Our TRANSITIVE model which implements this scheme is shown in Figure 2. Each mention  $i$  has been augmented with a single property node  $p_i \in \{1, \dots, k\}$ . The unary  $P_i$  factors encode prior knowledge about the setting of each  $p_i$ ; these factors may be hard (*I* will not refer to a plural entity), soft (such as a distribution over named entity types output by an NER tagger), or practically uniform (e.g. the last name Smith does not specify a particular gender).

To enforce agreement of a particular property, we require a mention to have the same property value as its antecedent. That is, for mentions  $i$  and  $j$ , if  $a_i = j$ , we want to ensure that  $p_i$  and  $p_j$  agree. We can achieve this with the following set of structural equality factors:

$$E_{i-j}(a_i, p_i, p_j) = 1 - \mathbb{I}[a_i = j \wedge p_i \neq p_j]$$

In words, this factor is zero if both  $a_i = j$  and  $p_i$  disagrees with  $p_j$ . These equality factors essentially provide a mechanism by which these priors  $P_i$  can influence the coreference decisions: if, for example, the factors  $P_i$  and  $P_j$  disagree very strongly, choosing  $a_i \neq j$  will be preferred in order to avoid forcing one of  $p_i$  or  $p_j$  to take an undesirable value. Moreover, note that although  $a_i$

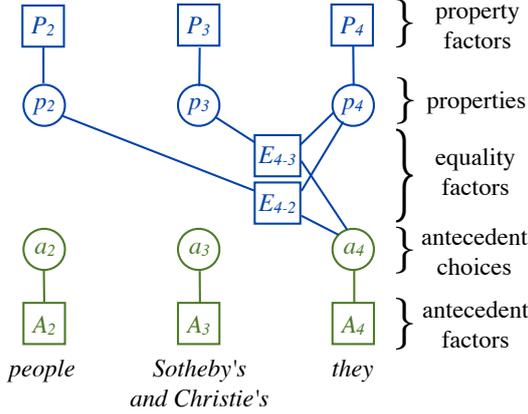


Figure 2: The factor graph for our TRANSITIVE coreference model. Each node  $a_i$  now has a property  $p_i$ , which is informed by its own unary factor  $P_i$ . In our example,  $a_4$  strongly indicates that mentions 2 and 4 are coreferent; the factor  $E_{4-2}$  then enforces equality between  $p_2$  and  $p_4$ , while the factor  $E_{4-3}$  has no effect.

only indicates a single antecedent, the transitive nature of the  $E$  factors forces  $p_i$  to agree with the  $p$  nodes of all other mentions likely to be in the same entity.

### 3.4 Property Projection

So far, our model as specified ensures agreement of our entity-level properties, but strictly enforcing agreement may not always be correct. Suppose that we are using named entity type as an entity-level property. Organizations and geo-political entities are two frequently confused and ambiguous tags, and in the gold-standard coreference chains it may be the case that a single chain contains instances of both. We might wish to learn that organizations and geo-political entities are “compatible” in the sense that we should forgive entities for containing both, but without losing the ability to reject a chain containing both organizations and people, for example.

To address these effects, we expand our model as indicated in Figure 3. As before, we have a set of properties  $p_i$  and agreement factors  $E_{ij}$ . On top of that, we introduce the notion of raw property values  $r_i \in \{1, \dots, k\}$  together with priors in the form of the  $R_i$  factors. The  $r_i$  and  $p_i$  could in principle have different domains, but for this work we take them to have the same domain. The  $P_i$  factors now have a new structure: they now represent a featurized projection of the  $r_i$  onto the  $p_i$ , which can now be thought of as “coreference-

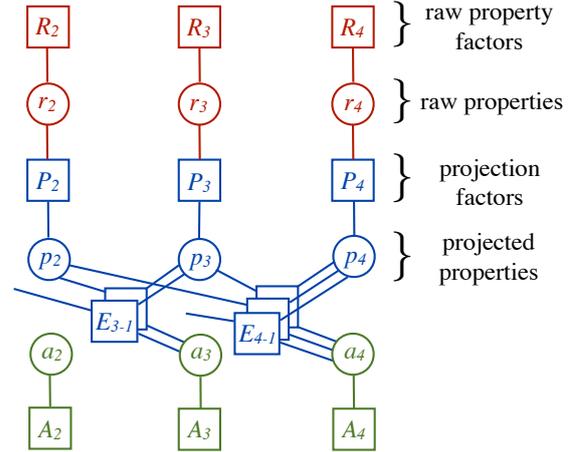


Figure 3: The complete factor graph for our TRANSITIVE coreference model. Compared to Figure 2, the  $R_i$  contain the raw cluster posteriors, and the  $P_i$  factors now project raw cluster values  $r_i$  into a set of “coreference-adapted” clusters  $p_i$  that are used as before. This projection allows mentions with different but compatible raw property values to coexist in the same coreference chain.

adapted” properties. The  $P_i$  factors are defined by  $P_i(p_i, r_i) \propto \exp(\mathbf{w}^T \mathbf{f}_P(p_i, r_i))$ , where  $\mathbf{f}_P$  is a feature vector over the projection of  $r_i$  onto  $p_i$ . While there are many possible choices of  $\mathbf{f}_P$ , we choose it to be an indicator of the values of  $p_i$  and  $r_i$ , so that we learn a fully-parameterized projection matrix.<sup>2</sup> The  $R_i$  are constant factors, and may come from an upstream model or some other source depending on the property being modeled.

Our description thus far has assumed that we are modeling only one type of property. In fact, we can use multiple properties for each mention by duplicating the  $r$  and  $p$  nodes and the  $R$ ,  $P$ , and  $E$  factors across each desired property. We index each of these by  $l \in \{1, \dots, m\}$  for each of  $m$  properties.

The final log-linear model is given by the following formula:

$$P(a|x) \propto \sum_{p,r} \left[ \left( \prod_{i,j,l} E_{l,i-j}(a_i, p_{li}, p_{lj}) \right) \left( \prod_{i,l} R_{li}(r_{li}) \right) \exp \left( \mathbf{w}^T \sum_i \left( \mathbf{f}_A(i, a_i, x) + \sum_l \mathbf{f}_P(p_{li}, r_{li}) \right) \right) \right]$$

where  $i$  and  $j$  range over mentions,  $l$  ranges over

<sup>2</sup>Initialized to zero (or small values), this matrix actually causes the transitive machinery to have no effect, since all posteriors over the  $p_i$  are flat and completely uninformative. Therefore, we regularize the weights of the indicators of  $p_i = r_i$  towards 1 and all other features towards 0 to give each raw cluster a preference for a distinct projected cluster.

each of  $m$  properties, and the outer sum indicates marginalization over all  $p$  and  $r$  variables.

## 4 Learning

Now that we have defined our model, we must decide how to train its weights  $\mathbf{w}$ . The first issue to address is one of the supervision provided. Our model traffics in sets of labels  $a$  which are more specified than gold coreference chains  $C$ , which give cluster membership for each mention but not antecedence. Let  $\mathcal{A}(C)$  be the set of labelings  $a$  that are consistent with a set of coreference chains  $C$ . For example, if  $C = \{\{1, 2, 3\}, \{4\}\}$ , then  $(\langle \text{new} \rangle, 1, 2, \langle \text{new} \rangle) \in \mathcal{A}(C)$  and  $(\langle \text{new} \rangle, 1, 1, \langle \text{new} \rangle) \in \mathcal{A}(C)$  but  $(\langle \text{new} \rangle, 1, \langle \text{new} \rangle, 3) \notin \mathcal{A}(C)$ , since this implies the chains  $C = \{\{1, 2\}, \{3, 4\}\}$

The most natural objective is a variant of standard conditional log-likelihood that treats the choice of  $a$  for the specified  $C$  as a latent variable to be marginalized out:

$$\ell(\mathbf{w}) = \sum_{i=1}^t \log \left( \sum_{a \in \mathcal{A}(C^i)} P(a|x^i) \right) \quad (1)$$

where  $(x^i, C^i)$  is the  $i$ th labeled training example. This optimizes for the 0-1 loss; however, we are much more interested in optimizing with respect to a coreference-specific loss function.

To this end, we will use softmax-margin (Gimpel and Smith, 2010), which augments the probability of each example with a term proportional to its loss, pushing the model to assign less mass to highly incorrect examples. We modify Equation 1 to use a new probability distribution  $P'$  instead of  $P$ , where  $P'(a|x^i) \propto P(a|x^i) \exp(l(a, C))$  and  $l(a, C)$  is a loss function. In order to perform inference efficiently,  $l(a, C)$  must decompose linearly across mentions:  $l(a, C) = \sum_{i=1}^n l(a_i, C)$ . Commonly-used coreference metrics such as MUC (Vilain et al., 1995) and  $B^3$  (Bagga and Baldwin, 1998) do not have this property, so we instead make use of a parameterized loss function that does and fit the parameters to give good performance. Specifically, we take

$$l(a, C) = \sum_{i=1}^n [c_1 \mathbb{I}(K_1(a_i, C)) + c_2 \mathbb{I}(K_2(a_i, C)) + c_3 \mathbb{I}(K_3(a_i, C))]$$

where  $c_1$ ,  $c_2$ , and  $c_3$  are real-valued weights,  $K_1$  denotes the event that  $a_i$  is falsely anaphoric when

it should be non-anaphoric,  $K_2$  denotes the event that  $a_i$  is falsely non-anaphoric when it should be anaphoric, and  $K_3$  denotes the event that  $a_i$  is correctly determined to be anaphoric but . These can be computed based on only  $a_i$  and  $C$ . By setting  $c_1$  low and  $c_2$  high relative to  $c_3$ , we can force the system to be less conservative about making anaphoricity decisions and achieve a better balance with the final coreference metrics.

Finally, we incorporate  $L_1$  regularization, giving us our final objective:

$$\ell(\mathbf{w}) = \sum_{i=1}^t \log \left( \sum_{a \in \mathcal{A}(C^i)} P'(a|x^i) \right) + \lambda \|\mathbf{w}\|_1$$

We optimize this objective using AdaGrad (Duchi et al., 2011); we found this to be faster and give higher performance than L-BFGS using  $L_2$  regularization (Liu and Nocedal, 1989). Note that because of the marginalization over  $\mathcal{A}(C^i)$ , even the objective for the BASIC model is not convex.

## 5 Inference

Inference in the BASIC model is straightforward. Given a set of weights  $\mathbf{w}$ , we can predict

$$\hat{a} = \arg \max_a P(a|x)$$

We then report the corresponding chains  $C(a)$  as the system output.<sup>3</sup> For learning, the gradient takes the standard form of the gradient of a log-linear model, a difference of expected feature counts under the gold annotation and under no annotation. This requires computing marginals  $P'(a_i|x)$  for each mention  $i$ , but because the model already factors this way, this step is easy.

The TRANSITIVE model is more complex. Exact inference is intractable due to the  $E$  factors that couple all of the  $a_i$  by way of the  $p_i$  nodes. However, we can compute approximate marginals for the  $a_i$ ,  $p_i$ , and  $r_i$  using belief propagation. BP has been effectively used on other NLP tasks (Smith and Eisner, 2008; Burkett and Klein, 2012), and is effective in cases such as this where the model is largely driven by non-loopy factors (here, the  $A_i$ ).

From marginals over each node, we can compute the necessary gradient and decode as before:

$$\hat{a} = \arg \max_a \hat{P}(a|x)$$

<sup>3</sup>One could use ILP-based decoding in the style of Finkel and Manning (2008) and Song et al. (2012) to attempt to explicitly find the optimal  $C$  with choice of  $a$  marginalized out, but we did not explore this option.

This corresponds to minimum-risk decoding with respect to the Hamming loss over antecedence predictions.

**Pruning.** The TRANSITIVE model requires instantiating a factor for each potential setting of each  $a_i$ . This factor graph grows quadratically in the size of the document, and even approximate inference becomes slow when a document contains over 200 mentions. Therefore, we use our BASIC model to prune antecedent choices for each  $a_i$  in order to reduce the size of the factor graph that we must instantiate. Specifically, we prune links between pairs of mentions that are of mention distance more than 100, as well as values for  $a_i$  that fall below a particular odds ratio threshold with respect to the best setting of that  $a_i$  in the BASIC model; that is, those for which

$$\log \left( \frac{P_{\text{BASIC}}(a_i|x)}{\max_j P_{\text{BASIC}}(a_i=j|x)} \right)$$

is below a cutoff  $\gamma$ .

## 6 Related Work

Our BASIC model is a mention-ranking approach resembling models used by Denis and Baldrige (2008) and Rahman and Ng (2009), though it is trained using a novel parameterized loss function. It is also similar to the MLN-JOINT(BF) model of Song et al. (2012), but we enforce the single-parent constraint at a deeper structural level, allowing us to treat non-anaphoricity symmetrically with coreference as in Denis and Baldrige (2007) and Stoyanov and Eisner (2012). The model of Fernandes et al. (2012) also uses the single-parent constraint structurally, but with learning via latent perceptron and ILP-based one-best decoding rather than logistic regression and BP-based marginal computation.

Our TRANSITIVE model is novel; while McCallum and Wellner (2004) proposed the idea of using attributes for mentions, they do not actually implement a model that does so. Other systems include entity-level information via handwritten rules (Raghunathan et al., 2010), induced rules (Yang et al., 2008), or features with learned weights (Luo et al., 2004; Rahman and Ng, 2011), but all of these systems freeze past coreference decisions in order to compute their entities.

Most similar to our entity-level approach is the system of Haghghi and Klein (2010), which

also uses approximate global inference; however, theirs is an unsupervised, generative system and they attempt to directly model multinomials over words in each mention. Their system could be extended to handle property information like we do, but our system has many other advantages, such as freedom from a pre-specified list of entity types, the ability to use multiple input clusterings, and discriminative projection of clusters.

## 7 Experiments

We use the datasets, experimental setup, and scoring program from the CoNLL 2011 shared task (Pradhan et al., 2011), based on the OntoNotes corpus (Hovy et al., 2006). We use the standard automatic parses and NER tags for each document. Our mentions are those output by the system of Lee et al. (2011); we also use their postprocessing to remove appositives, predicate nominatives, and singletons before evaluation. For each experiment, we report MUC (Vilain et al., 1995),  $B^3$  (Bagga and Baldwin, 1998), and CEAF<sub>e</sub> (Luo, 2005), as well as their average.

**Parameter settings.** We take the regularization constant  $\lambda = 0.001$  and the parameters of our surrogate loss  $(c_1, c_2, c_3) = (0.15, 2.5, 1)$  for all models.<sup>4</sup> All models are trained for 20 iterations. We take the pruning threshold  $\gamma = -2$ .

### 7.1 Systems

Besides our BASIC and TRANSITIVE systems, we evaluate a strictly pairwise system that incorporates property information by way of indicator features on the current mention’s most likely property value and the proposed antecedent’s most likely property value. We call this system PAIRPROPERTY; it is simply the BASIC system with an expanded feature set.

Furthermore, we compare against a LEFT-TORIGHT entity-level system like that of Rahman and Ng (2009).<sup>5</sup> Decoding now operates in a sequential fashion, with BASIC features computed as before and entity features computed for each mention based on the coreference decisions made thus far. Following Rahman and Ng (2009), features for each property indicate whether the cur-

<sup>4</sup>Additional tuning of these hyper parameters did not significantly improve any of the models under any of the experimental conditions.

<sup>5</sup>Unfortunately, their publicly-available system is closed-source and performs poorly on the CoNLL shared task dataset, so direct comparison is difficult.

rent mention agrees with no mentions in the antecedent cluster, at least one mention, over half of the mentions, or all of the mentions; antecedent clusters of size 1 or 2 fire special-cased features. These additional features beyond those in Rahman and Ng (2009) were helpful, but more involved conjunction schemes and fine-grained features were not. During training, entity features of both the gold and the prediction are computed using the Viterbi clustering of preceding mentions under the current model parameters.<sup>6</sup>

All systems are run in a two-pass manner: first, the BASIC model is run, then antecedent choices are pruned, then our second-round model is trained from scratch on the pruned data.<sup>7</sup>

## 7.2 Noisy Oracle Features

We first evaluate our model’s ability to exploit synthetic entity-level properties. For this experiment, mention properties are derived from corrupted oracle information about the true underlying coreference cluster. Each coreference cluster is assumed to have one underlying value for each of  $m$  coreference properties, each taking values over a domain  $D$ . Mentions then sample distributions over  $D$  from a Dirichlet distribution peaked around the true underlying value.<sup>8</sup> These posteriors are taken as the  $R_i$  for the TRANSITIVE model.

We choose this setup to reflect two important properties of entity-level information: first, that it may come from a variety of disparate sources, and second, that it may be based on the determinations of upstream models which produce posteriors naturally. A strength of our model is that it can accept such posteriors as input, naturally making use of this information in a model-based way.

Table 1 shows development results averaged across ten train-test splits with  $m = 3$  properties, each taking one of  $|D| = 5$  values. We emphasize that these parameter settings give fairly weak oracle information: a document may have hundreds of clusters, so even in the absence of noise these oracle properties do not have high dis-

<sup>6</sup>Using gold entities for training as in Rahman and Ng (2009) resulted in a lower-performing system.

<sup>7</sup>We even do this for the BASIC model, since we found that performance of the pruned and retrained model was generally higher.

<sup>8</sup>Specifically, the distribution used is a Dirichlet with  $\alpha = 3.5$  for the true underlying cluster and  $\alpha = 1$  for other values, chosen so that 25% of samples from the distribution did not have the correct mode. Though these parameters affect the quality of the oracle information, varying them did not change the relative performance of the different models.

NOISY ORACLE				
	MUC	$B^3$	CEAF <sub>e</sub>	Avg.
BASIC	61.96	70.66	47.30	59.97
PAIRPROPERTY	66.31	72.68	49.08	62.69
LEFTTORIGHT	66.49	73.14	49.46	63.03
TRANSITIVE	<b>67.37</b>	<b>74.05</b>	<b>49.68</b>	<b>63.70</b>

Table 1: CoNLL metric scores for our four different systems incorporating noisy oracle data. This information helps substantially in all cases. Both entity-level models outperform the PAIRPROPERTY model, but we observe that the TRANSITIVE model is more effective than the LEFTTORIGHT model at using this information.

criminating power. Still, we see that all models are able to benefit from incorporating this information; however, our TRANSITIVE model outperforms both the PAIRPROPERTY model and the LEFTTORIGHT model. There are a few reasons for this: first, our model is able to directly use soft posteriors, so it is able to exploit the fact that more peaked samples from the Dirichlet are more likely to be correct. Moreover, our model can propagate information backwards in a document as well as forwards, so the effects of noise can be more easily mitigated. By contrast, in the LEFTTORIGHT model, if the first or second mention in a cluster has the wrong property value, features indicating high levels of property agreement will not fire on the next few mentions in those clusters.

## 7.3 Phi Features

As we have seen, our TRANSITIVE model can exploit high-quality entity-level features. How does it perform using real features that have been proposed for entity-level coreference?

Here, we use hard phi feature determinations extracted from the system of Lee et al. (2011). Named-entity type and animacy are both computed based on the output of a named-entity tagger, while number and gender use the dataset of Bergsma and Lin (2006). Once this information is determined, the PAIRPROPERTY and LEFTTORIGHT systems can compute features over it directly. In the TRANSITIVE model, each of the  $R_i$  factors places  $\frac{3}{4}$  of its mass on the determined label and distributes the remainder uniformly among the possible options.

Table 2 shows results when adding entity-level phi features on top of our BASIC pairwise system (which already contains pairwise features) and on top of an ablated BASIC system without pairwise

PHI FEATURES				
	MUC	$B^3$	CEAF <sub>e</sub>	Avg.
BASIC	61.96	70.66	47.30	59.97
LEFTTORIGHT	61.34	70.41	<b>47.64</b>	59.80
TRANSITIVE	<b>62.66</b>	<b>70.92</b>	46.88	<b>60.16</b>
PHI FEATURES (ABLATED BASIC)				
BASIC-PHI	59.45	69.21	46.02	58.23
PAIRPROPERTY	61.88	70.66	47.14	59.90
LEFTTORIGHT	61.42	70.53	<b>47.49</b>	59.81
TRANSITIVE	<b>62.23</b>	<b>70.78</b>	46.74	<b>59.92</b>

Table 2: CoNLL metric scores for our systems incorporating phi features. Our standard BASIC system already includes phi features, so no results are reported for PAIRPROPERTY. Here, our TRANSITIVE system does not give substantial improvement on the averaged metric. Over a baseline which does not include phi features, all systems are able to incorporate them comparably.

phi features. Our entity-level systems successfully captures phi features when they are not present in the baseline, but there is only slight benefit over pairwise incorporation, a result which has been noted previously (Luo et al., 2004).

#### 7.4 Clustering Features

Finally, we consider mention properties derived from unsupervised clusterings; these properties are designed to target semantic properties of nominals that should behave more like the oracle features than the phi features do.

We consider clusterings that take as input pairs  $(n, r)$  of a noun head  $n$  and a string  $r$  which contains the semantic role of  $n$  (or some approximation thereof) conjoined with its governor. Two different algorithms are used to cluster these pairs: a NAIVEBAYES model, where  $c$  generates  $n$  and  $r$ , and a CONDITIONAL model, where  $c$  is generated conditioned on  $r$  and then  $n$  is generated from  $c$ . Parameters for each can be learned with the expectation maximization (EM) algorithm (Dempster et al., 1977), with symmetry broken by a small amount of random noise at initialization.

Similar models have been used to learn sub-categorization information (Rooth et al., 1999) or properties of verb argument slots (Yao et al., 2011). We choose this kind of clustering for its relative simplicity and because it allows pronouns to have more informed properties (from their verbal context) than would be possible using a model that makes type-level decisions about nominals only. Though these specific cluster features are novel to coreference, previous work has used similar

CLUSTERS				
	MUC	$B^3$	CEAF <sub>e</sub>	Avg.
BASIC	61.96	70.66	47.30	59.97
PAIRPROPERTY	62.88	70.71	<b>47.45</b>	60.35
LEFTTORIGHT	61.98	70.19	45.77	59.31
TRANSITIVE	<b>63.34</b>	<b>70.89</b>	46.88	<b>60.37</b>

Table 3: CoNLL metric scores for our systems incorporating clustering features. These features are equally effectively incorporated by our PAIRPROPERTY system and our TRANSITIVE system.

government officials court authorities ...	ARG0:said ARG0:say ARG0:found ARG0:announced ...	way law agreement plan ...	ARG1:signed ARG1:announced ARG1:set ARG1:approved ...	...
attack problems attacks charges ...	ARG1:cause ARG2:following ARG1:reported ARG1:filed ...	prices shares index rates ...	ARG1:rose ARG1:fell ARG1:cut ARG1:closed ...	

Figure 4: Examples of clusters produced by the NAIVEBAYES model on SRL-tagged data with pronouns discarded.

types of fine-grained semantic class information (Hendrickx and Daelemans, 2007; Ng, 2007; Rahman and Ng, 2010). Other approaches incorporate information from other sources (Ponzetto and Strube, 2006) or compute heuristic scores for real-valued features based on a large corpus or the web (Dagan and Itai, 1990; Yang et al., 2005; Bansal and Klein, 2012).

We use four different clusterings in our experiments, each with twenty clusters: dependency-parse-derived NAIVEBAYES clusters, semantic-role-derived CONDITIONAL clusters, SRL-derived NAIVEBAYES clusters generating a NOVERB token when  $r$  cannot be determined, and SRL-derived NAIVEBAYES clusters with all pronoun tuples discarded. Examples of the latter clusters are shown in Figure 4. Each clustering is learned for 30 iterations of EM over English Gigaword (Graff et al., 2007), parsed with the Berkeley Parser (Petrov et al., 2006) and with SRL determined by Senna (Collobert et al., 2011).

Table 3 shows results of modeling these cluster properties. As in the case of oracle features, the PAIRPROPERTY and LEFTTORIGHT systems use the modes of the cluster posteriors, and the TRANSITIVE system uses the posteriors directly as the  $R_i$ . We see comparable performance from incorporating features in both an entity-level framework and a pairwise framework, though the TRANSI-

	MUC			$B^3$			CEAF <sub>e</sub>			Avg.
	Prec.	Rec.	$F_1$	Prec.	Rec.	$F_1$	Prec.	Rec.	$F_1$	$F_1$
BASIC	69.99	55.59	61.96	80.96	62.69	70.66	41.37	55.21	47.30	59.97
STANFORD	61.49	59.59	60.49	74.60	68.25	71.28	47.57	49.45	48.49	60.10
NOISY ORACLE										
PAIRPROPERTY	76.49	58.53	66.31	84.98	63.48	72.68	41.84	59.36	49.08	62.69
LEFTTORIGHT	76.92	58.55	66.49	85.68	63.81	73.14	42.07	60.01	49.46	63.03
TRANSITIVE	76.48	60.20	*67.37	84.84	65.69	*74.05	42.89	59.01	*49.68	63.70
PHI FEATURES										
LEFTTORIGHT	69.77	54.73	61.34	81.40	62.04	70.41	41.49	55.92	47.64	59.80
TRANSITIVE	70.27	56.54	*62.66	79.81	63.82	*70.92	41.17	54.44	46.88	60.16
PHI FEATURES (ABLATED BASIC)										
BASIC-PHI	67.04	53.41	59.45	78.93	61.63	69.21	40.40	53.46	46.02	58.23
PAIRPROPERTY	70.24	55.31	61.88	81.10	62.60	70.66	41.04	55.38	47.14	59.90
LEFTTORIGHT	69.94	54.75	61.42	81.38	62.23	70.53	41.29	55.87	47.49	59.81
TRANSITIVE	70.06	55.98	*62.23	79.92	63.52	70.78	40.90	54.52	46.74	59.92
CLUSTERS										
PAIRPROPERTY	71.77	55.95	62.88	81.76	62.30	70.71	40.98	56.35	47.45	60.35
LEFTTORIGHT	69.75	54.82	61.39	81.48	62.29	70.60	41.62	55.89	47.71	59.90
TRANSITIVE	71.54	56.83	*63.34	80.55	63.31	*70.89	40.77	55.14	46.88	60.37

Table 4: CoNLL metric scores averaged across ten different splits of the training set for each experiment. We include precision, recall, and  $F_1$  for each metric for completeness. Starred  $F_1$  values on the individual metrics for the TRANSITIVE system are significantly better than all other results in the same block at the  $p = 0.01$  level according to a bootstrap resampling test.

	MUC			$B^3$			CEAF <sub>e</sub>			Avg.
	Prec.	Rec.	$F_1$	Prec.	Rec.	$F_1$	Prec.	Rec.	$F_1$	$F_1$
BASIC	68.84	56.08	61.81	77.60	61.40	68.56	38.25	50.57	43.55	57.97
PAIRPROPERTY	70.90	56.26	62.73	78.95	60.79	68.69	37.69	51.92	43.67	58.37
LEFTTORIGHT	68.84	55.56	61.49	78.64	61.03	68.72	38.97	51.74	44.46	58.22
TRANSITIVE	70.62	58.06	*63.73	76.93	62.24	68.81	38.00	50.40	43.33	58.62
STANFORD	60.91	62.13	61.51	70.61	67.75	69.15	45.79	44.55	45.16	58.61

Table 5: CoNLL metric scores for our best systems (including clustering features) on the CoNLL blind test set, reported in the same manner as Table 4.

TIVE system appears to be more effective than the LEFTTORIGHT system.

## 7.5 Final Results

Table 4 shows expanded results on our development sets for the different types of entity-level information we considered. We also show in Table 5 the results of our system on the CoNLL test set, and see that it performs comparably to the Stanford coreference system (Lee et al., 2011). Here, our TRANSITIVE system provides modest improvements over all our other systems.

Based on Table 4, our TRANSITIVE system appears to do better on MUC and  $B^3$  than on CEAF<sub>e</sub>. However, we found no simple way to change the relative performance characteristics of our various systems; notably, modifying the parameters of the loss function mentioned in Section 4 or changing it entirely did not trade off these three metrics but merely increased or decreased them in lockstep. Therefore, the TRANSITIVE system actually substantially improves over our baselines and is not

merely trading off metrics in a way that could be easily reproduced through other means.

## 8 Conclusion

In this work, we presented a novel coreference architecture that can both take advantage of standard pairwise features as well as use transitivity to enforce coherence of decentralized entity-level properties within coreference clusters. Our transitive system is more effective at using properties than a pairwise system and a previous entity-level system, and it achieves performance comparable to that of the Stanford coreference resolution system, the winner of the CoNLL 2011 shared task.

## Acknowledgments

This work was partially supported by BBN under DARPA contract HR0011-12-C-0014, by an NSF fellowship for the first author, and by a Google fellowship for the second. Thanks to the anonymous reviewers for their insightful comments.

## References

- Amit Bagga and Breck Baldwin. 1998. Algorithms for Scoring Coreference Chains. In *Proceedings of the Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*.
- Mohit Bansal and Dan Klein. 2012. Coreference Semantics from Web Features. In *Proceedings of the Association for Computational Linguistics*.
- Eric Bengtson and Dan Roth. 2008. Understanding the Value of Features for Coreference Resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Shane Bergsma and Dekang Lin. 2006. Bootstrapping Path-Based Pronoun Resolution. In *Proceedings of the Conference on Computational Linguistics and the Association for Computational Linguistics*.
- David Burkett and Dan Klein. 2012. Fast Inference in Phrase Extraction Models with Belief Propagation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, November.
- Ido Dagan and Alon Itai. 1990. Automatic Processing of Large Corpora for the Resolution of Anaphora References. In *Proceedings of the Conference on Computational Linguistics - Volume 3*.
- Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.
- Pascal Denis and Jason Baldridge. 2007. Joint Determination of Anaphoricity and Coreference Resolution using Integer Programming. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.
- Pascal Denis and Jason Baldridge. 2008. Specialized Models and Ranking for Coreference Resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, July.
- Eraldo Rezende Fernandes, Cícero Nogueira dos Santos, and Ruy Luiz Milidiú. 2012. Latent Structure Perceptron with Feature Induction for Unrestricted Coreference Resolution. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Conference on Computational Natural Language Learning - Shared Task*.
- Jenny Rose Finkel and Christopher D. Manning. 2008. Enforcing Transitivity in Coreference Resolution. In *Proceedings of the Association for Computational Linguistics: Short Papers*.
- Kevin Gimpel and Noah A. Smith. 2010. Softmax-Margin CRFs: Training Log-Linear Models with Cost Functions. In *Proceedings of the North American Chapter for the Association for Computational Linguistics*.
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English Gigaword Third Edition. Linguistic Data Consortium, Catalog Number LDC2007T07.
- Aria Haghighi and Dan Klein. 2010. Coreference Resolution in a Modular, Entity-Centered Model. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.
- Iris Hendrickx and Walter Daelemans, 2007. *Adding Semantic Information: Unsupervised Clusters for Coreference Resolution*.
- Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: the 90% solution. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Short Papers*.
- Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford’s Multi-Pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task. In *Proceedings of the Conference on Computational Natural Language Learning: Shared Task*.
- Dong C. Liu and Jorge Nocedal. 1989. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming*, 45(3):503–528, December.
- Xiaoqiang Luo, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A Mention-Synchronous Coreference Resolution Algorithm Based on the Bell Tree. In *Proceedings of the Association for Computational Linguistics*.
- Xiaoqiang Luo. 2005. On Coreference Resolution Performance Metrics. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Andrew McCallum and Ben Wellner. 2004. Conditional Models of Identity Uncertainty with Application to Noun Coreference. In *Proceedings of Advances in Neural Information Processing Systems*.
- George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38:39–41.
- Vincent Ng. 2007. Semantic class induction and coreference resolution. In *Proceedings of the Association for Computational Linguistics*.

- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the Conference on Computational Linguistics and the Association for Computational Linguistics*.
- Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting Semantic Role Labeling, WordNet and Wikipedia for Coreference Resolution. In *Proceedings of the North American Chapter of the Association of Computational Linguistics*.
- Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 Shared Task: Modeling Unrestricted Coreference in OntoNotes. In *Proceedings of the Conference on Computational Natural Language Learning: Shared Task*.
- Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A Multi-Pass Sieve for Coreference Resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Altaf Rahman and Vincent Ng. 2009. Supervised Models for Coreference Resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Altaf Rahman and Vincent Ng. 2010. Inducing Fine-Grained Semantic Classes via Hierarchical and Collective Classification. In *Proceedings of the International Conference on Computational Linguistics*.
- Altaf Rahman and Vincent Ng. 2011. Narrowing the Modeling Gap: A Cluster-Ranking Approach to Coreference Resolution. *Journal of Artificial Intelligence Research*, 40(1):469–521, January.
- Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a Semantically Annotated Lexicon via EM-Based Clustering. In *Proceedings of the Association for Computational Linguistics*.
- David A. Smith and Jason Eisner. 2008. Dependency Parsing by Belief Propagation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Yang Song, Jing Jiang, Wayne Xin Zhao, Sujian Li, and Houfeng Wang. 2012. Joint Learning for Coreference Resolution with Markov Logic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Veselin Stoyanov and Jason Eisner. 2012. Easy-first Coreference Resolution. In *Proceedings of the International Conference on Computational Linguistics*.
- Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010. Coreference Resolution with Reconcile. In *Proceedings of the Association for Computational Linguistics: Short Papers*.
- Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A Model-Theoretic Coreference Scoring Scheme. In *Proceedings of the Conference on Message Understanding*.
- Xiaofeng Yang, Jian Su, and Chew Lim Tan. 2005. Improving Pronoun Resolution Using Statistics-Based Semantic Compatibility Information. In *Proceedings of the Association for Computational Linguistics*.
- Xiaofeng Yang, Jian Su, Jun Lang, Chew L. Tan, Ting Liu, and Sheng Li. 2008. An Entity-Mention Model for Coreference Resolution with Inductive Logic Programming. In *Proceedings of the Association for Computational Linguistics*.
- Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured Relation Discovery Using Generative Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

# Chinese Parsing Exploiting Characters

Meishan Zhang<sup>†</sup>, Yue Zhang<sup>‡\*</sup>, Wanxiang Che<sup>†</sup>, Ting Liu<sup>†</sup>

<sup>†</sup>Research Center for Social Computing and Information Retrieval  
Harbin Institute of Technology, China

{mszhang, car, tliu}@ir.hit.edu.cn

<sup>‡</sup>Singapore University of Technology and Design  
yue\_zhang@sutd.edu.sg

## Abstract

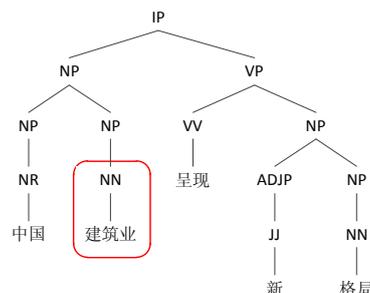
Characters play an important role in the Chinese language, yet computational processing of Chinese has been dominated by word-based approaches, with leaves in syntax trees being words. We investigate Chinese parsing from the character-level, extending the notion of phrase-structure trees by annotating internal structures of words. We demonstrate the importance of character-level information to Chinese processing by building a joint segmentation, part-of-speech (POS) tagging and phrase-structure parsing system that integrates character-structure features. Our joint system significantly outperforms a state-of-the-art word-based baseline on the standard CTB5 test, and gives the best published results for Chinese parsing.

## 1 Introduction

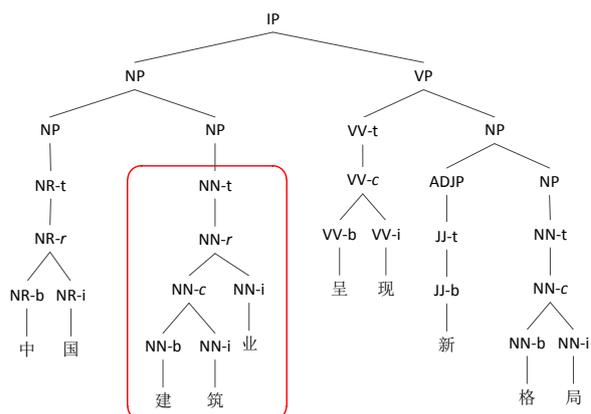
Characters play an important role in the Chinese language. They act as basic phonetic, morpho-syntactic and semantic units in a Chinese sentence. Frequently-occurring character sequences that express certain meanings can be treated as words, while most Chinese words have syntactic structures. For example, Figure 1(b) shows the structure of the word “建筑业 (construction and building industry)”, where the characters “建 (construction)” and “筑 (building)” form a coordination, and modify the character “业 (industry)”.

However, computational processing of Chinese is typically based on words. Words are treated as the atomic units in syntactic parsing, machine translation, question answering and other NLP tasks. Manually annotated corpora, such as the Chinese Treebank (CTB) (Xue et al., 2005), usually have words as the basic syntactic elements

\*Email correspondence.



(a) CTB-style word-based syntax tree for “中国 (China) 建筑业 (architecture industry) 呈现 (show) 新 (new) 格局 (pattern)”.



(b) character-level syntax tree with hierarchal word structures for “中 (middle) 国 (nation) 建 (construction) 筑 (building) 业 (industry) 呈 (present) 现 (show) 新 (new) 格 (style) 局 (situation)”.

Figure 1: Word-based and character-level phrase-structure trees for the sentence “中国建筑业呈现新格局 (China’s architecture industry shows new patterns)”, where “l”, “r”, “c” denote the directions of head characters (see section 2).

(Figure 1(a)). This form of annotation does not give character-level syntactic structures for words, a source of linguistic information that is more fundamental and less sparse than atomic words.

In this paper, we investigate Chinese syntactic parsing with character-level information by extending the notation of phrase-structure

(constituent) trees, adding recursive structures of characters for words. We manually annotate the structures of 37,382 words, which cover the entire CTB5. Using these annotations, we transform CTB-style constituent trees into character-level trees (Figure 1(b)). Our word structure corpus, together with a set of tools to transform CTB-style trees into character-level trees, is released at <https://github.com/zhangmeishan/wordstructures>. Our annotation work is in line with the work of Vadas and Curran (2007) and Li (2011), which provide extended annotations of Penn Treebank (PTB) noun phrases and CTB words (on the morphological level), respectively.

We build a character-based Chinese parsing model to parse the character-level syntax trees. Given an input Chinese sentence, our parser produces its character-level syntax trees (Figure 1(b)). With richer information than word-level trees, this form of parse trees can be useful for all the aforementioned Chinese NLP applications.

With regard to task of parsing itself, an important advantage of the character-level syntax trees is that they allow word segmentation, part-of-speech (POS) tagging and parsing to be performed jointly, using an efficient CKY-style or shift-reduce algorithm. Luo (2003) exploited this advantage by adding flat word structures without manually annotation to CTB trees, and building a generative character-based parser. Compared to a pipeline system, the advantages of a joint system include reduction of error propagation, and the integration of segmentation, POS tagging and syntax features. With hierarchical structures and head character information, our annotated words are more informative than flat word structures, and hence can bring further improvements to phrase-structure parsing.

To analyze word structures in addition to phrase structures, our character-based parser naturally performs joint word segmentation, POS tagging and parsing jointly. Our model is based on the discriminative shift-reduce parser of Zhang and Clark (2009; 2011), which is a state-of-the-art word-based phrase-structure parser for Chinese. We extend their shift-reduce framework, adding more transition actions for word segmentation and POS tagging, and defining novel features that capture character information. Even when trained using character-level syntax trees with flat word structures, our joint parser outperforms a strong pipelined baseline that consists of a state-of-the-

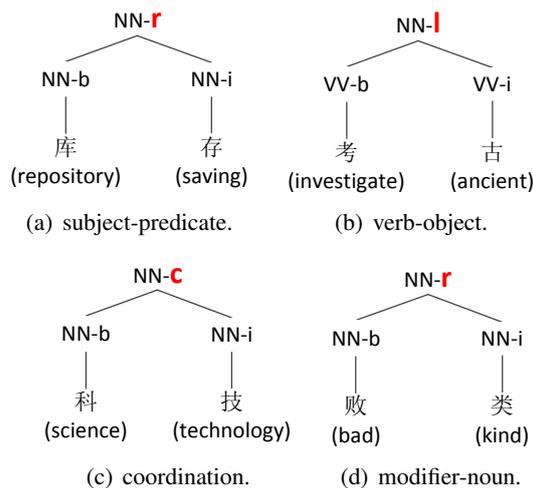


Figure 2: Inner word structures of “库存 (repository)”, “考古 (archaeology)”, “科技 (science and technology)” and “败类 (degenerate)”.

art joint segmenter and POS tagger, and our baseline word-based parser. Our word annotations lead to further improvements to the joint system, especially for phrase-structure parsing accuracy.

Our parser work falls in line with recent work of joint segmentation, POS tagging and parsing (Hatori et al., 2012; Li and Zhou, 2012; Qian and Liu, 2012). Compared with related work, our model gives the best published results for joint segmentation and POS tagging, as well as joint phrase-structure parsing on standard CTB5 evaluations. With linear-time complexity, our parser is highly efficient, processing over 30 sentences per second with a beam size of 16. An open release of the parser is freely available at <http://sourceforge.net/projects/zpar/>, version 0.6.

## 2 Word Structures and Syntax Trees

The Chinese language is a character-based language. Unlike alphabetical languages, Chinese characters convey meanings, and the meaning of most Chinese words takes roots in their character. For example, the word “计算机 (computer)” is composed of the characters “计 (count)”, “算 (calculate)” and “机 (machine)”. An informal name of “computer” is “电脑”, which is composed of “电 (electronic)” and “脑 (brain)”.

Chinese words have internal structures (Xue, 2001; Ma et al., 2012). The way characters interact within words can be similar to the way words interact within phrases. Figure 2 shows the structures of the four words “库存 (repository)”, “考古

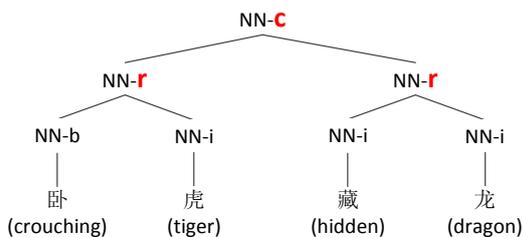


Figure 3: Character-level word structure of “卧虎藏龙 (crouching tiger hidden dragon)”.

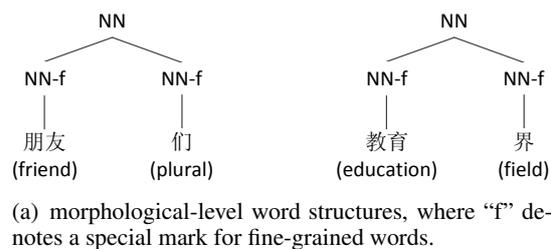
(archaeology)”, “科技 (science and technology)” and “败类 (degenerate)”, which demonstrate four typical syntactic structures of two-character words, including subject-predicate, verb-object, coordination and modifier-noun structures. Multi-character words can also have recursive syntactic structures. Figure 3 illustrates the structure of the word “卧虎藏龙 (crouching tiger hidden dragon)”, which is composed of two subwords “卧虎 (crouching tiger)” and “藏龙 (hidden dragon)”, both having a modifier-noun structure.

The meaning of characters can be a useful source of information for computational processing of Chinese, and some recent work has started to exploit this information. Zhang and Clark (2010) found that the first character in a Chinese word is a useful indicator of the word’s POS. They made use of this information to help joint word segmentation and POS tagging.

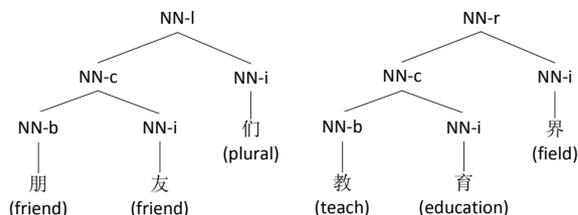
Li (2011) studied the morphological structures of Chinese words, showing that 35% percent of the words in CTB5 can be treated as having morphemes. Figure 4(a) illustrates the morphological structures of the words “朋友们 (friends)” and “教育界 (educational world)”, in which the characters “们 (plural)” and “界 (field)” can be treated as suffix morphemes. They studied the influence of such morphology to Chinese dependency parsing (Li and Zhou, 2012).

The aforementioned work explores the influence of particular types of characters to Chinese processing, yet not the full potentials of complete word structures. We take one step further in this line of work, annotating the full syntactic structures of 37,382 Chinese words in the form of Figure 2 and Figure 3. Our annotation covers the entire vocabulary of CTB5. In addition to difference in coverage (100% vs 35%), our annotation is structurally more informative than that of Li (2011), as illustrated in Figure 4(b).

Our annotations are binarized recursive word



(a) morphological-level word structures, where “f” denotes a special mark for fine-grained words.



(b) character-level word structures.

Figure 4: Comparison between character-level and morphological-level word structures.

structures. For each word or subword, we specify its POS and head direction. We use “l”, “r” and “c” to indicate the “left”, “right” and “coordination” head directions, respectively. The “coordination” direction is mostly used in coordination structures, while a very small number of transliteration words, such as “奥巴马 (Obama)” and “洛杉矶 (Los Angeles)”, have flat structures, and we use “coordination” for their left binarization. For leaf characters, we follow previous work on word segmentation (Xue, 2003; Ng and Low, 2004), and use “b” and “i” to indicate the beginning and non-beginning characters of a word, respectively.

The vast majority of words do not have structural ambiguities. However, the structures of some words may vary according to different POS. For example, “制服” means “dominate” when it is tagged as a verb, of which the head is the left character; the same word means “uniform dress” when tagged as a noun, of which the head is the right character. Thus the input of the word structure annotation is a word together with its POS. The annotation work was conducted by three persons, with one person annotating the entire corpus, and the other two checking the annotations.

Using our annotations, we can extend CTB-style syntax trees (Figure 1(a)) into *character-level* trees (Figure 1(b)). In particular, we mark the original nodes that represent POS tags in CTB-style trees with “-t”, and insert our word structures as unary subnodes of the “-t” nodes. For the rest of the paper, we refer to the “-t” nodes as *full-word nodes*, all nodes above full-word nodes as *phrase*

nodes, and all nodes below full-word nodes as *subword nodes*.

Our character-level trees contain additional syntactic information, which are potentially useful to Chinese processing. For example, the head characters of words can be populated up to phrase-level nodes, and serve as an additional source of information that is less sparse than head words. In this paper, we build a parser that yields character-level trees from raw character sequences. In addition, we use this parser to study the effects of our annotations to character-based statistical Chinese parsing, showing that they are useful in improving parsing accuracies.

### 3 Character-based Chinese Parsing

To produce character-level trees for Chinese NLP tasks, we develop a character-based parsing model, which can jointly perform word segmentation, POS tagging and *phrase-structure* parsing. To our knowledge, this is the first work to develop a transition-based system that jointly performs the above three tasks. Trained using annotated word structures, our parser also analyzes the internal structures of Chinese words.

Our character-based Chinese parsing model is based on the work of Zhang and Clark (2009), which is a transition-based model for lexicalized constituent parsing. They use a beam-search decoder so that the transition action sequence can be globally optimized. The averaged perceptron with early-update (Collins and Roark, 2004) is used to train the model parameters. Their transition system contains four kinds of actions: (1) *SHIFT*, (2) *REDUCE-UNARY*, (3) *REDUCE-BINARY* and (4) *TERMINATE*. The system can provide binarized CFG trees in Chomsky Norm Form, and they present a reversible conversion procedure to map arbitrary CFG trees into binarized trees.

In this work, we remain consistent with their work, using the head-finding rules of Zhang and Clark (2008), and the same binarization algorithm.<sup>1</sup> We apply the same beam-search algorithm for decoding, and employ the averaged perceptron with early-update to train our model.

We make two extensions to their work to enable joint segmentation, POS tagging and phrase-structure parsing from the character level. First, we modify the actions of the transition system for

<sup>1</sup>We use a left-binarization process for flat word structures that contain more than two characters.

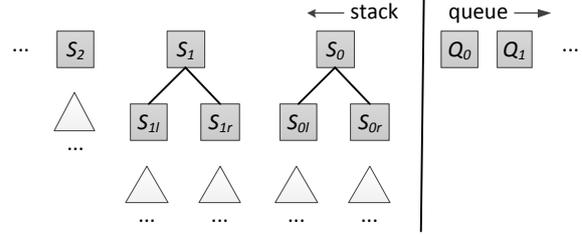


Figure 5: A state in a transition-based model.

parsing the inner structures of words. Second, we extend the feature set for our parsing problem.

#### 3.1 The Transition System

In a transition-based system, an input sentence is processed in a linear left-to-right pass, and the output is constructed by a state-transition process. We learn a model for scoring the transition  $A_i$  from one state  $ST_i$  to the next  $ST_{i+1}$ . As shown in Figure 5, a state  $ST$  consists of a stack  $S$  and a queue  $Q$ , where  $S = (\dots, S_1, S_0)$  contains partially constructed parse trees, and  $Q = (Q_0, Q_1, \dots, Q_{n-j}) = (c_j, c_{j+1}, \dots, c_n)$  is the sequence of input characters that have not been processed. The candidate transition action  $A$  at each step is defined as follows:

- *SHIFT-SEPARATE* ( $t$ ): remove the head character  $c_j$  from  $Q$ , pushing a subword node  $\frac{S'.2}{c_j}$  onto  $S$ , assigning  $S'.t = t$ . Note that the parse tree  $S_0$  must correspond to a full-word or a phrase node, and the character  $c_j$  is the first character of the next word. The argument  $t$  denotes the POS of  $S'$ .
- *SHIFT-APPEND*: remove the head character  $c_j$  from  $Q$ , pushing a subword node  $\frac{S'}{c_j}$  onto  $S$ .  $c_j$  will eventually be combined with all the subword nodes on top of  $S$  to form a word, and thus we must have  $S'.t = S_0.t$ .
- *REDUCE-SUBWORD* ( $d$ ): pop the top two nodes  $S_0$  and  $S_1$  off  $S$ , pushing a new subword node  $\frac{S'}{S_1 S_0}$  onto  $S$ . The argument  $d$  denotes the head direction of  $S'$ , of which the value can be “left”, “right” or “coordination”.<sup>3</sup> Both  $S_0$  and  $S_1$  must be subword nodes and  $S'.t = S_0.t = S_1.t$ .

<sup>2</sup>We use this notation for a compact representation of a tree node, where the numerator represents a father node, and the denominator represents the children.

<sup>3</sup>For the head direction “coordination”, we extract the head character from the left node.

Category	Feature templates	When to Apply
Structure features	$S_{0ntl} S_{0nwl} S_{1ntl} S_{1nwl} S_{2ntl} S_{2nwl} S_{3ntl} S_{3nwl}$ , $Q_0c Q_1c Q_2c Q_3c Q_0c \cdot Q_1c Q_1c \cdot Q_2c Q_2c \cdot Q_3c$ , $S_{0l}twl S_{0r}twl S_{0u}twl S_{1l}twl S_{1r}twl S_{1u}twl$ , $S_{0nw} \cdot S_{1nw} S_{0nw} \cdot S_{1nl} S_{0nl} \cdot S_{1nw} S_{0nl} \cdot S_{1nl}$ , $S_{0nw} \cdot Q_0c S_{0nl} \cdot Q_0c S_{1nw} \cdot Q_0c S_{1nl}Q_0c$ , $S_{0nl} \cdot S_{1nl} \cdot S_{2nl} S_{0nw} \cdot S_{1nl} \cdot S_{2nl} S_{0nl} \cdot S_{1nw} \cdot S_{2nl} S_{0nl} \cdot S_{1nl} \cdot S_{2nw}$ , $S_{0nw} \cdot S_{1nl} \cdot Q_0c S_{0nl} \cdot S_{1nw} \cdot Q_0c S_{0nl} \cdot S_{1nl} \cdot Q_0c$ , $S_{0ncl} S_{0nctl} S_{0nct} S_{1ncl} S_{1nctl} S_{1nct}$ , $S_{2ncl} S_{2nctl} S_{2nct} S_{3ncl} S_{3nctl} S_{3nct}$ , $S_{0nc} \cdot S_{1nc} S_{0ncl} \cdot S_{1nl} S_{0nl} \cdot S_{1ncl} S_{0ncl} \cdot S_{1ncl}$ , $S_{0nc} \cdot Q_0c S_{0nl} \cdot Q_0c S_{1nc} \cdot Q_0c S_{1nl} \cdot Q_0c$ , $S_{0nc} \cdot S_{1nc} \cdot Q_0c S_{0ncl} \cdot S_{1nc} \cdot Q_0c \cdot Q_1c$	All
	$\text{start}(S_0w) \cdot \text{start}(S_1w) \text{ start}(S_0w) \cdot \text{end}(S_1w)$ , $\text{indict}(S_1wS_0w) \cdot \text{len}(S_1wS_0w) \text{ indict}(S_1wS_0w, S_0t) \cdot \text{len}(S_1wS_0w)$	REDUCE-SUBWORD
String features	$t_{-1} \cdot t_0 \quad t_{-2} \cdot t_{-1}t_0 \quad w_{-1} \cdot t_0 \quad c_0 \cdot t_0 \quad \text{start}(w_{-1}) \cdot t_0 \quad c_{-1} \cdot c_0 \cdot t_{-1} \cdot t_0$ ,	SHIFT-SEPARATE
	$w_{-1} \quad w_{-2} \cdot w_{-1} \quad w_{-1}$ , where $\text{len}(w_{-1}) = 1 \quad \text{end}(w_{-1}) \cdot c_0$ , $\text{start}(w_{-1}) \cdot \text{len}(w_{-1}) \quad \text{end}(w_{-1}) \cdot \text{len}(w_{-1}) \quad \text{start}(w_{-1}) \cdot \text{end}(w_{-1})$ , $w_{-1} \cdot c_0 \quad \text{end}(w_{-2}) \cdot w_{-1} \quad \text{start}(w_{-1}) \cdot c_0 \quad \text{end}(w_{-2}) \cdot \text{end}(w_{-1})$ , $w_{-1} \cdot \text{len}(w_{-2}) \quad w_{-2} \cdot \text{len}(w_{-1}) \quad w_{-1} \cdot t_{-1} \quad w_{-1} \cdot t_{-2} \quad w_{-1} \cdot t_{-1} \cdot c_0$ , $w_{-1} \cdot t_{-1} \cdot \text{end}(w_{-2}) \quad c_{-2} \cdot c_{-1} \cdot c_0 \cdot t_{-1}$ , where $\text{len}(w_{-1}) = 1 \quad \text{end}(w_{-1}) \cdot t_{-1}$ , $c \cdot t_{-1} \cdot \text{end}(w_{-1})$ , where $c \in w_{-1}$ and $c \neq \text{end}(w_{-1})$	REDUCE-WORD
	$c_0 \cdot t_{-1} \quad c_{-1} \cdot c_0 \quad \text{start}(w_{-1}) \cdot c_0t_{-1} \quad c_{-1} \cdot c_0 \cdot t_{-1}$	SHIFT-APPEND

Table 1: Feature templates for the character-level parser. The function  $\text{start}(\cdot)$ ,  $\text{end}(\cdot)$  and  $\text{len}(\cdot)$  denote the first character, the last character and the length of a word, respectively.

- REDUCE-WORD: pop the top node  $S_0$  off  $S$ , pushing a full-word node  $\frac{S'}{S_0}$  onto  $S$ . This reduce action generates a full-word node from  $S_0$ , which must be a subword node.
- REDUCE-BINARY ( $d, l$ ): pop the top two nodes  $S_0$  and  $S_1$  off  $S$ , pushing a binary phrase node  $\frac{S'}{S_1 S_0}$  onto  $S$ . The argument  $l$  denotes the constituent label of  $S'$ , and the argument  $d$  specifies the lexical head direction of  $S'$ , which can be either “left” or “right”. Both  $S_0$  and  $S_1$  must be a full-word node or a phrase node.
- REDUCE-UNARY ( $l$ ): pop the top node  $S_0$  off  $S$ , pushing a unary phrase node  $\frac{S'}{S_0}$  onto  $S$ .  $l$  denotes the constituent label of  $S'$ .
- TERMINATE: mark parsing complete.

Compared to set of actions in our baseline transition-based phrase-structure parser, we have made three major changes. First, we split the original SHIFT action into SHIFT-SEPARATE ( $t$ ) and SHIFT-APPEND, which jointly perform the word segmentation and POS tagging tasks. Second, we add an extra REDUCE-SUBWORD ( $d$ ) operation, which is used for parsing the inner struc-

tures of words. Third, we add REDUCE-WORD, which applies a unary rule to mark a completed subword node as a full-word node. The new node corresponds to a unary “-t” node in Figure 1(b).

### 3.2 Features

Table 1 shows the feature templates of our model. The feature set consists of two categories: (1) structure features, which encode the structural information of subwords, full-words and phrases. (2) string features, which encode the information of neighboring characters and words.

For the structure features, the symbols  $S_0, S_1, S_2, S_3$  represent the top four nodes on the stack;  $Q_0, Q_1, Q_2, Q_3$  denote the first four characters in the queue;  $S_{0l}, S_{0r}, S_{0u}$  represent the left, right child for a binary branching  $S_0$ , and the single child for a unary branching  $S_0$ , respectively;  $S_{1l}, S_{1r}, S_{1u}$  represent the left, right child for a binary branching  $S_1$ , and the single child for a unary branching  $S_1$ , respectively;  $n$  represents the type for a node; it is a binary value that indicates whether the node is a subword node;  $c, w, t$  and  $l$  represent the head character, word (or subword), POS tag and constituent label of a node, respectively. The structure features are mostly taken

from the work of Zhang and Clark (2009). The feature templates in bold are novel, are designed to encode head character information. In particular, the **indict** function denotes whether a word is in a tag dictionary, which is collected by extracting all multi-character subwords that occur more than five times in the training corpus.

For string features,  $c_0$ ,  $c_{-1}$  and  $c_{-2}$  represent the current character and its previous two characters, respectively;  $w_{-1}$  and  $w_{-2}$  represent the previous two words to the current character, respectively;  $t_0$ ,  $t_{-1}$  and  $t_{-2}$  represent the POS tags of the current word and the previous two words, respectively. The string features are used for word segmentation and POS tagging, and are adapted from a state-of-the-art joint segmentation and tagging model (Zhang and Clark, 2010).

In summary, our character-based parser contains the word-based features of constituent parser presented in Zhang and Clark (2009), the word-based and shallow character-based features of joint word segmentation and POS tagging presented in Zhang and Clark (2010), and additionally the deep character-based features that encode word structure information, which are the first presented by this paper.

## 4 Experiments

### 4.1 Setting

We conduct our experiments on the CTB5 corpus, using the standard split of data, with sections 1–270,400–931 and 1001–1151 for training, sections 301–325 for system development, and sections 271–300 for testing. We apply the same pre-processing step as Harper and Huang (2011), so that the non-terminal yield unary chains are collapsed to single unary rules.

Since our model can jointly process word segmentation, POS tagging and phrase-structure parsing, we evaluate our model for the three tasks, respectively. For word segmentation and POS tagging, standard metrics of word precision, recall and F-score are used, where the tagging accuracy is the joint accuracy of word segmentation and POS tagging. For phrase-structure parsing, we use the standard `parseval` evaluation metrics on bracketing precision, recall and F-score. As our constituent trees are based on characters, we follow previous work and redefine the boundary of a constituent span by its start and end characters. In addition, we evaluate the performance of word

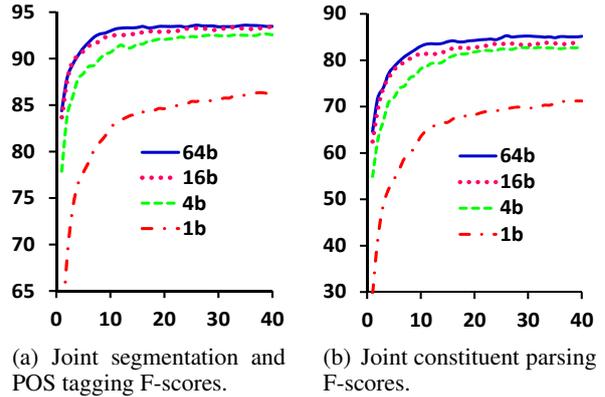


Figure 6: Accuracies against the training epoch for joint segmentation and tagging as well as joint phrase-structure parsing using beam sizes 1, 4, 16 and 64, respectively.

structures, using the word precision, recall and F-score metrics. A word structure is correct only if the word and its internal structure are both correct.

### 4.2 Development Results

Figure 6 shows the accuracies of our model using different beam sizes with respect to the training epoch. The performance of our model increases as the beam size increases. The amount of increases becomes smaller as the size of the beam grows larger. Tested using gcc 4.7.2 and Fedora 17 on an Intel Core i5-3470 CPU (3.20GHz), the decoding speeds are 318.2, 98.0, 30.3 and 7.9 sentences per second with beam size 1, 4, 16 and 64, respectively. Based on this experiment, we set the beam size 64 for the rest of our experiments.

The character-level parsing model has the advantage that deep character information can be extracted as features for parsing. For example, the head character of a word is exploited in our model. We conduct feature ablation experiments to evaluate the effectiveness of these features. We find that the parsing accuracy decreases about 0.6% when the head character related features (the bold feature templates in Table 1) are removed, which demonstrates the usefulness of these features.

### 4.3 Final Results

In this section, we present the final results of our model, and compare it to two baseline systems, a pipelined system and a joint system that is trained with automatically generated flat words structures.

The baseline pipelined system consists of the joint segmentation and tagging model proposed by

	Task	P	R	F
Pipeline	Seg	97.35	98.02	97.69
	Tag	93.51	94.15	93.83
	Parse	81.58	82.95	82.26
Flat word structures	Seg	97.32	98.13	97.73
	Tag	94.09	94.88	94.48
	Parse	83.39	83.84	83.61
Annotated word structures	Seg	97.49	98.18	<b>97.84</b>
	Tag	94.46	95.14	<b>94.80</b>
	Parse	84.42	84.43	<b>84.43</b>
	WS	94.02	94.69	94.35

Table 2: Final results on test corpus.

Zhang and Clark (2010), and the phrase-structure parsing model of Zhang and Clark (2009). Both models give state-of-the-art performances, and are freely available.<sup>4</sup> The model for joint segmentation and POS tagging is trained with a 16-beam, since it achieves the best performance. The phrase-structure parsing model is trained with a 64-beam. We train the parsing model using the automatically generated POS tags by 10-way jackknifing, which gives about 1.5% increases in parsing accuracy when tested on automatic segmented and POS tagged inputs.

The joint system trained with flat word structures serves to test the effectiveness of our joint parsing system over the pipelined baseline, since flat word structures do not contain additional sources of information over the baseline. It is also used to test the usefulness of our annotation in improving parsing accuracy.

Table 2 shows the final results of our model and the two baseline systems on the test data. We can see that both character-level joint models outperform the pipelined system; our model with annotated word structures gives an improvement of 0.97% in tagging accuracy and 2.17% in phrase-structure parsing accuracy. The results also demonstrate that the annotated word structures are highly effective for syntactic parsing, giving an absolute improvement of 0.82% in phrase-structure parsing accuracy over the joint model with flat word structures.

Row “WS” in Table 2 shows the accuracy of hierarchical word-structure recovery of our joint system. This figure can be useful for high-level applications that make use of character-level trees by

<sup>4</sup><http://sourceforge.net/projects/zpar/>, version 0.5.

our parser, as it reflects the capability of our parser in analyzing word structures. In particular, the performance of parsing OOV word structure is an important metric of our parser. The recall of OOV word structures is 60.43%, while if we do not consider the influences of segmentation and tagging errors, counting only the correctly segmented and tagged words, the recall is 87.96%.

#### 4.4 Comparison with Previous Work

In this section, we compare our model to previous systems on the performance of joint word segmentation and POS tagging, and the performance of joint phrase-structure parsing.

Table 3 shows the results. Kruengkrai+ ’09 denotes the results of Kruengkrai et al. (2009), which is a lattice-based joint word segmentation and POS tagging model; Sun ’11 denotes a subword based stacking model for joint segmentation and POS tagging (Sun, 2011), which uses a dictionary of idioms; Wang+ ’11 denotes a semi-supervised model proposed by Wang et al. (2011), which additionally uses the Chinese Gigaword Corpus; Li ’11 denotes a generative model that can perform word segmentation, POS tagging and phrase-structure parsing jointly (Li, 2011); Li+ ’12 denotes a unified dependency parsing model that can perform joint word segmentation, POS tagging and dependency parsing (Li and Zhou, 2012); Li ’11 and Li+ ’12 exploited annotated morphological-level word structures for Chinese; Hatori+ ’12 denotes an incremental joint model for word segmentation, POS tagging and dependency parsing (Hatori et al., 2012); they use external dictionary resources including HowNet Word List and page names from the Chinese Wikipedia; Qian+ ’12 denotes a joint segmentation, POS tagging and parsing system using a unified framework for decoding, incorporating a word segmentation model, a POS tagging model and a phrase-structure parsing model together (Qian and Liu, 2012); their word segmentation model is a combination of character-based model and word-based model. Our model achieved the best performance on both joint segmentation and tagging as well as joint phrase-structure parsing.

Our final performance on constituent parsing is by far the best that we are aware of for the Chinese data, and even better than some state-of-the-art models with gold segmentation. For example, the un-lexicalized PCFG model of Petrov and Klein

System	Seg	Tag	Parse
Kruengkrai+ '09	97.87	93.67	–
Sun '11	98.17*	94.02*	–
Wang+ '11	98.11*	94.18*	–
Li '11	97.3	93.5	79.7
Li+ '12	97.50	93.31	–
Hatori+ '12	<b>98.26*</b>	94.64*	–
Qian+ '12	97.96	93.81	82.85
Ours pipeline	97.69	93.83	82.26
Ours joint flat	97.73	94.48	83.61
Ours joint annotated	97.84	<b>94.80</b>	<b>84.43</b>

Table 3: Comparisons of our final model with state-of-the-art systems, where “\*” denotes that external dictionary or corpus has been used.

(2007) achieves 83.45%<sup>5</sup> in parsing accuracy on the test corpus, and our pipeline constituent parsing model achieves 83.55% with gold segmentation. They are lower than the performance of our character-level model, which is 84.43% without gold segmentation. The main differences between word-based and character-level parsing models are that character-level model can exploit character features. This further demonstrates the effectiveness of characters in Chinese parsing.

## 5 Related Work

Recent work on using the internal structure of words to help Chinese processing gives important motivations to our work. Zhao (2009) studied character-level dependencies for Chinese word segmentation by formalizing segmentation task in a dependency parsing framework. Their results demonstrate that annotated word dependencies can be helpful for word segmentation. Li (2011) pointed out that the word’s internal structure is very important for Chinese NLP. They annotated morphological-level word structures, and a unified generative model was proposed to parse the Chinese morphological and phrase-structures. Li and Zhou (2012) also exploited the morphological-level word structures for Chinese dependency parsing. They proposed a unified transition-based model to parse the morphological and dependency structures of a Chinese sentence in a unified framework. The morphological-level word struc-

<sup>5</sup>We rerun the parser and evaluate it using the publicly-available code on <http://code.google.com/p/berkeleyparser> by ourselves, since we have a preprocessing step for the CTB5 corpus.

tures concern only prefixes and suffixes, which cover only 35% of entire words in CTB. According to their results, the final performances of their model on word segmentation and POS tagging are below the state-of-the-art joint segmentation and POS tagging models. Compared to their work, we consider the character-level word structures for Chinese parsing, presenting a unified framework for segmentation, POS tagging and phrase-structure parsing. We can achieve improved segmentation and tagging performance.

Our character-level parsing model is inspired by the work of Zhang and Clark (2009), which is a transition-based model with a beam-search decoder for word-based constituent parsing. Our work is based on the shift-reduce operations of their work, while we introduce additional operations for segmentation and POS tagging. By such an extension, our model can include all the features in their work, together with the features for segmentation and POS tagging. In addition, we propose novel features related to word structures and interactions between word segmentation, POS tagging and word-based constituent parsing.

Luo (2003) was the first work to introduce the character-based syntax parsing. They use it as a joint framework to perform Chinese word segmentation, POS tagging and syntax parsing. They exploit a generative maximum entropy model for character-based constituent parsing, and find that POS information is very useful for Chinese word segmentation, but high-level syntactic information seems to have little effect on segmentation. Compared to their work, we use a transition-based discriminative model, which can benefit from large amounts of flexible features. In addition, instead of using flat structures, we manually annotate hierarchical tree structures of Chinese words for converting word-based constituent trees into character-based constituent trees.

Hatori et al. (2012) proposed the first joint work for the word segmentation, POS tagging and dependency parsing. They used a single transition-based model to perform the three tasks. Their work demonstrates that a joint model can improve the performance of the three tasks, particularly for POS tagging and dependency parsing. Qian and Liu (2012) proposed a joint decoder for word segmentation, POS tagging and word-based constituent parsing, although they trained models for the three tasks separately. They reported better

performances when using a joint decoder. In our work, we employ a single character-based discriminative model to perform segmentation, POS tagging and phrase-structure parsing jointly, and study the influence of annotated word structures.

## 6 Conclusions and Future Work

We studied the internal structures of more than 37,382 Chinese words, analyzing their structures as the recursive combinations of characters. Using these word structures, we extended the CTB into character-level trees, and developed a character-based parser that builds such trees from raw character sequences. Our character-based parser performs segmentation, POS tagging and parsing simultaneously, and significantly outperforms a pipelined baseline. We make both our annotations and our parser available online.

In summary, our contributions include:

- We annotated the internal structures of Chinese words, which are potentially useful to character-based studies of Chinese NLP. We extend CTB-style constituent trees into character-level trees using our annotations.
- We developed a character-based parsing model that can produce our character-level constituent trees. Our parser jointly performs word segmentation, POS tagging and syntactic parsing.
- We investigated the effectiveness of our joint parser over pipelined baseline, and the effectiveness of our annotated word structures in improving parsing accuracies.

Future work includes investigations of our parser and annotations on Chinese NLP tasks.

## Acknowledgments

This work was supported by National Natural Science Foundation of China (NSFC) via grant 61133012, the National “863” Major Projects via grant 2011AA01A207, the National “863” Leading Technology Research Project via grant 2012AA011102, and SRG ISTD 2012 038 from Singapore University of Technology and Design.

## References

- Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL’04), Main Volume*, pages 111–118, Barcelona, Spain, July.
- Mary Harper and Zhongqiang Huang. 2011. Chinese statistical parsing. *Handbook of Natural Language Processing and Machine Translation*.
- Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun’ichi Tsujii. 2012. Incremental joint approach to word segmentation, pos tagging, and dependency parsing in chinese. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1045–1053, Jeju Island, Korea, July. Association for Computational Linguistics.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun’ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 513–521, Suntec, Singapore, August. Association for Computational Linguistics.
- Zhongguo Li and Guodong Zhou. 2012. Unified dependency parsing of chinese morphological and syntactic structures. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1445–1454, Jeju Island, Korea, July. Association for Computational Linguistics.
- Zhongguo Li. 2011. Parsing the internal structure of words: A new paradigm for chinese word segmentation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1405–1414, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Xiaoqiang Luo. 2003. A maximum entropy Chinese character-based parser. In Michael Collins and Mark Steedman, editors, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 192–199.
- Jianqiang Ma, Chunyu Kit, and Dale Gerdemann. 2012. Semi-automatic annotation of chinese word structure. In *Proceedings of the Second CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 9–17, Tianjin, China, December. Association for Computational Linguistics.
- Hwee Tou Ng and Jin Kiat Low. 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 277–284, Barcelona, Spain, July. Association for Computational Linguistics.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language*

- Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York, April. Association for Computational Linguistics.
- Xian Qian and Yang Liu. 2012. Joint chinese word segmentation, pos tagging and parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 501–511, Jeju Island, Korea, July. Association for Computational Linguistics.
- Weiwei Sun. 2011. A stacked sub-word model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1385–1394, Portland, Oregon, USA, June. Association for Computational Linguistics.
- David Vadas and James Curran. 2007. Adding noun phrase structure to the penn treebank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 240–247, Prague, Czech Republic, June. Association for Computational Linguistics.
- Yiou Wang, Jun’ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving chinese word segmentation and pos tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 309–317, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.
- Nianwen Xue. 2001. *Defining and Automatically Identifying Words in Chinese*. Ph.D. thesis, University of Delaware.
- Nianwen Xue. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1).
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Yue Zhang and Stephen Clark. 2009. Transition-based parsing of the chinese treebank using a global discriminative model. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT’09)*, pages 162–171, Paris, France, October. Association for Computational Linguistics.
- Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 843–852, Cambridge, MA, October. Association for Computational Linguistics.
- Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.
- Hai Zhao. 2009. Character-level dependencies in chinese: Usefulness and learning. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 879–887, Athens, Greece, March. Association for Computational Linguistics.

# A Transition-Based Dependency Parser Using a Dynamic Parsing Strategy

**Francesco Sartorio**  
Department of  
Information Engineering  
University of Padua, Italy  
sartorio@dei.unipd.it

**Giorgio Satta**  
Department of  
Information Engineering  
University of Padua, Italy  
satta@dei.unipd.it

**Joakim Nivre**  
Department of  
Linguistics and Philology  
Uppsala University, Sweden  
joakim.nivre@lingfil.uu.se

## Abstract

We present a novel transition-based, greedy dependency parser which implements a flexible mix of bottom-up and top-down strategies. The new strategy allows the parser to postpone difficult decisions until the relevant information becomes available. The novel parser has a  $\sim 12\%$  error reduction in unlabeled attachment score over an arc-eager parser, with a slow-down factor of 2.8.

## 1 Introduction

Dependency-based methods for syntactic parsing have become increasingly popular during the last decade or so. This development is probably due to many factors, such as the increased availability of dependency treebanks and the perceived usefulness of dependency structures as an interface to downstream applications, but a very important reason is also the high efficiency offered by dependency parsers, enabling web-scale parsing with high throughput. The most efficient parsers are greedy transition-based parsers, which only explore a single derivation for each input and relies on a locally trained classifier for predicting the next parser action given a compact representation of the derivation history, as pioneered by Yamada and Matsumoto (2003), Nivre (2003), Attardi (2006), and others. However, while these parsers are capable of processing tens of thousands of tokens per second with the right choice of classifiers, they are also known to perform slightly below the state-of-the-art because of search errors and subsequent error propagation (McDonald and Nivre, 2007), and recent research on transition-based dependency parsing has therefore explored different ways of improving their accuracy.

The most common approach is to use beam search instead of greedy decoding, in combination

with a globally trained model that tries to minimize the loss over the entire sentence instead of a locally trained classifier that tries to maximize the accuracy of single decisions (given no previous errors), as first proposed by Zhang and Clark (2008). With these methods, transition-based parsers have reached state-of-the-art accuracy for a number of languages (Zhang and Nivre, 2011; Bohnet and Nivre, 2012). However, the drawback with this approach is that parsing speed is proportional to the size of the beam, which means that the most accurate transition-based parsers are not nearly as fast as the original greedy transition-based parsers. Another line of research tries to retain the efficiency of greedy classifier-based parsing by instead improving the way in which classifiers are learned from data. While the classical approach limits training data to parser states that result from oracle predictions (derived from a treebank), these novel approaches allow the classifier to explore states that result from its own (sometimes erroneous) predictions (Choi and Palmer, 2011; Goldberg and Nivre, 2012).

In this paper, we explore an orthogonal approach to improving the accuracy of transition-based parsers, without sacrificing their advantage in efficiency, by introducing a new type of transition system. While all previous transition systems assume a static parsing strategy with respect to top-down and bottom-up processing, our new system allows a dynamic strategy for ordering parsing decisions. This has the advantage that the parser can postpone difficult decisions until the relevant information becomes available, in a way that is not possible in existing transition systems. A second advantage of dynamic parsing is that we can extend the feature inventory of previous systems. Our experiments show that these advantages lead to significant improvements in parsing accuracy, compared to a baseline parser that uses the arc-eager transition system of Nivre (2003), which is one of the most

widely used static transition systems.

## 2 Static vs. Dynamic Parsing

The notions of bottom-up and top-down parsing strategies do not have a general mathematical definition; they are instead specified, often only informally, for individual families of grammar formalisms. In the context of dependency parsing, a parsing strategy is called purely **bottom-up** if every dependency  $h \rightarrow d$  is constructed only after all dependencies of the form  $d \rightarrow i$  have been constructed. Here  $h \rightarrow d$  denotes a dependency with  $h$  the head node and  $d$  the dependent node. In contrast, a parsing strategy is called purely **top-down** if  $h \rightarrow d$  is constructed before any dependency of the form  $d \rightarrow i$ .

If we consider transition-based dependency parsing (Nivre, 2008), the purely bottom-up strategy is implemented by the arc-standard model of Nivre (2004). After building a dependency  $h \rightarrow d$ , this model immediately removes from its stack node  $d$ , preventing further attachment of dependents to this node. A second popular parser, the arc-eager model of Nivre (2003), instead adopts a mixed strategy. In this model, a dependency  $h \rightarrow d$  is constructed using a purely bottom-up strategy if it represents a left-arc, that is, if the dependent  $d$  is placed to the left of the head  $h$  in the input string. In contrast, if  $h \rightarrow d$  represents a right-arc (defined symmetrically), then this dependency is constructed before any right-arc  $d \rightarrow i$  (top-down) but after any left-arc  $d \rightarrow i$  (bottom-up).

What is important to notice about the above transition-based parsers is that the adopted parsing strategies are **static**. By this we mean that each dependency is constructed according to some *fixed* criterion, depending on structural conditions such as the fact that the dependency represents a left or a right arc. This should be contrasted with **dynamic** parsing strategies in which several parsing options are *simultaneously* available for the dependencies being constructed.

In the context of left-to-right, transition-based parsers, dynamic strategies are attractive for several reasons. One argument is related to the well-known PP-attachment problem, illustrated in Figure 1. Here we have to choose whether to attach node P as a dependent of V (arc  $\alpha_2$ ) or else as a dependent of N1 (arc  $\alpha_3$ ). The purely bottom-up arc-standard model has to take a decision as soon as N1 is placed into the stack. This is so

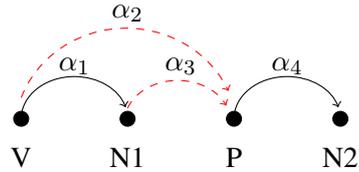


Figure 1: PP-attachment example, with dashed arcs identifying two alternative choices.

because the construction of  $\alpha_1$  excludes  $\alpha_3$  from the search space, while the alternative decision of shifting P into the stack excludes  $\alpha_2$ . This is bad, because the information about the correct attachment could come from the lexical content of node P. The arc-eager model performs slightly better, since it can delay the decision up to the point in which  $\alpha_1$  has been constructed and P is read from the buffer. However, at this point it must make a commitment and either construct  $\alpha_3$  or pop N1 from the stack (implicitly committing to  $\alpha_2$ ) before N2 is read from the buffer. In contrast with this scenario, in the next sections we implement a dynamic parsing strategy that allows a transition system to decide between the attachments  $\alpha_2$  and  $\alpha_3$  after it has seen all of the four nodes V, N1, P and N2.

Other additional advantages of dynamic parsing strategies with respect to static strategies are related to the increase in the feature inventory that we apply to parser states, and to the increase of spurious ambiguity. However, these arguments are more technical than the PP-attachment argument above, and will be discussed later.

## 3 Dependency Parser

In this section we present a novel transition-based parser for projective dependency trees, implementing a dynamic parsing strategy.

### 3.1 Preliminaries

For non-negative integers  $i$  and  $j$  with  $i \leq j$ , we write  $[i, j]$  to denote the set  $\{i, i+1, \dots, j\}$ . When  $i > j$ ,  $[i, j]$  is the empty set.

We represent an input sentence as a string  $w = w_0 \cdots w_n$ ,  $n \geq 1$ , where token  $w_0$  is a special root symbol and, for each  $i \in [1, n]$ , token  $w_i = (i, a_i, t_i)$  encodes a lexical element  $a_i$  and a part-of-speech tag  $t_i$  associated with the  $i$ -th word in the sentence.

A **dependency tree** for  $w$  is a directed, ordered tree  $T_w = (V_w, A_w)$ , where  $V_w = \{w_i \mid i \in$

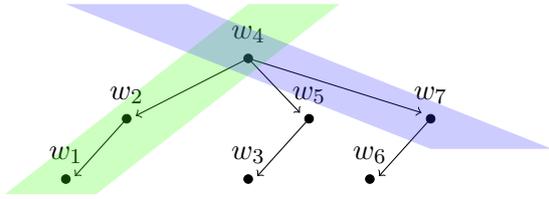


Figure 2: A dependency tree with left spine  $\langle w_4, w_2, w_1 \rangle$  and right spine  $\langle w_4, w_7 \rangle$ .

$[0, n]$  is the set of nodes, and  $A_w \subseteq V_w \times V_w$  is the set of arcs. Arc  $(w_i, w_j)$  encodes a dependency  $w_i \rightarrow w_j$ . A sample dependency tree (excluding  $w_0$ ) is displayed in Figure 2. If  $(w_i, w_j) \in A_w$  for  $j < i$ , we say that  $w_j$  is a left child of  $w_i$ ; a right child is defined in a symmetrical way.

The **left spine** of  $T_w$  is an ordered sequence  $\langle u_1, \dots, u_p \rangle$  with  $p \geq 1$  and  $u_i \in V_w$  for  $i \in [1, p]$ , consisting of all nodes in a descending path from the root of  $T_w$  taking the leftmost child node at each step. More formally,  $u_1$  is the root node of  $T_w$  and  $u_i$  is the leftmost child of  $u_{i-1}$ , for  $i \in [2, p]$ . The **right spine** of  $T_w$  is defined symmetrically; see again Figure 2. Note that the left and the right spines share the root node and no other node.

### 3.2 Basic Idea

Transition-based dependency parsers use a stack data structure, where each stack element is associated with a tree spanning some (contiguous) substring of the input  $w$ . The parser can combine two trees  $T$  and  $T'$  through attachment operations, called left-arc or right-arc, under the condition that  $T$  and  $T'$  appear at the two topmost positions in the stack. Crucially, only the roots of  $T$  and  $T'$  are available for attachment; see Figure 3(a).

In contrast, a stack element in our parser records the *entire* left spine and right spine of the associated tree. This allows us to extend the inventory of the attachment operations of the parser by including the attachment of tree  $T$  as a dependent of any node in the left or in the right spine of a second tree  $T'$ , provided that this does not violate projectivity.<sup>1</sup> See Figure 3(b) for an example.

The new parser implements a mix of bottom-up and top-down strategies, since after any of the attachments in Figure 3(b) is performed, additional dependencies can still be created for the root of  $T$ . Furthermore, the new parsing strategy is clearly dy-

<sup>1</sup>A dependency tree for  $w$  is projective if every subtree has a contiguous yield in  $w$ .

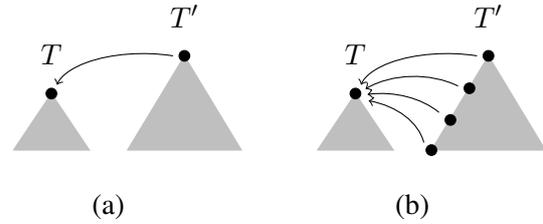


Figure 3: Left-arc attachment of  $T$  to  $T'$  in case of (a) standard transition-based parsers and (b) our parser.

namic, due to the free choice in the timing for these attachments. The new strategy is more powerful than the strategy of the arc-eager model, since we can use top-down parsing at left arcs, which is not allowed in arc-eager parsing, and we do not have the restrictions of parsing right arcs ( $h \rightarrow d$ ) before the attachment of right dependents at node  $d$ .

To conclude this section, let us resume our discussion of the PP-attachment example in Figure 1. We observe that the new parsing strategy allows the construction of a tree  $T'$  consisting of the only dependency  $V \rightarrow N1$  and a tree  $T$ , placed at the right of  $T'$ , consisting of the only dependency  $P \rightarrow N2$ . Since the right spine of  $T'$  consists of nodes  $V$  and  $N1$ , we can freely choose between attachment  $V \rightarrow P$  and attachment  $N1 \rightarrow P$ . Note that this is done after we have seen node  $N2$ , as desired.

### 3.3 Transition-based Parser

We assume the reader is familiar with the formal framework of transition-based dependency parsing originally introduced by Nivre (2003); see Nivre (2008) for an introduction. To keep the notation at a simple level, we only discuss here the unlabeled version of our parser; however, a labeled extension is used in §5 for our experiments.

Our transition-based parser uses a **stack** data structure to store partial parses for the input string  $w$ . We represent the stack as an ordered sequence  $\sigma = [\sigma_d, \dots, \sigma_1]$ ,  $d \geq 0$ , of stack elements, with the topmost element placed at the right. When  $d = 0$ , we have the empty stack  $\sigma = []$ . Sometimes we use the vertical bar to denote the append operator for  $\sigma$ , and write  $\sigma = \sigma' | \sigma_1$  to indicate that  $\sigma_1$  is the topmost element of  $\sigma$ .

A **stack element** is a pair

$$\sigma_k = (\langle u_{k,1}, \dots, u_{k,p} \rangle, \langle v_{k,1}, \dots, v_{k,q} \rangle)$$

where the ordered sequences  $\langle u_{k,1}, \dots, u_{k,p} \rangle$  and

$\langle v_{k,1}, \dots, v_{k,q} \rangle$  are the left and the right spines, respectively, of the tree associated with  $\sigma_k$ . Recall that  $u_{k,1} = v_{k,1}$ , since the root node of the associated tree is shared by the two spines.

The parser also uses a **buffer** to store the portion of the input string still to be processed. We represent the buffer as an ordered sequence  $\beta = [w_i, \dots, w_n]$ ,  $i \geq 0$ , of tokens from  $w$ , with the first element placed at the left. Note that  $\beta$  always represents a (non-necessarily proper) suffix of  $w$ . When  $i > n$ , we have the empty buffer  $\beta = []$ . Sometimes we use the vertical bar to denote the append operator for  $\beta$ , and write  $\beta = w_i | \beta'$  to indicate that  $w_i$  is the first token of  $\beta$ ; consequently, we have  $\beta' = [w_{i+1}, \dots, w_n]$ .

When processing  $w$ , the parser reaches several states, technically called configurations. A **configuration** of the parser relative to  $w$  is a triple  $c = (\sigma, \beta, A)$ , where  $\sigma$  and  $\beta$  are a stack and a buffer, respectively, and  $A \subseteq V_w \times V_w$  is a set of arcs. The initial configuration for  $w$  is  $([], [w_0, \dots, w_n], \emptyset)$ . The set of terminal configurations consists of all configurations of the form  $([\sigma_1], [], A)$ , where  $\sigma_1$  is associated with a tree having root  $w_0$ , that is,  $u_{1,1} = v_{1,1} = w_0$ , and  $A$  is any set of arcs.

The core of a transition-based parser is the set of its transitions. Each **transition** is a binary relation defined over the set of configurations of the parser. Since the set of configurations is infinite, a transition is infinite as well, when viewed as a set. However, transitions can always be specified by some finite means. Our parser uses three types of transitions, defined in what follows.

- **SHIFT**, or **sh** for short. This transition removes the first node from the buffer and pushes into the stack a new element, consisting of the left and right spines of the associated tree. More formally

$$(\sigma, w_i | \beta, A) \vdash_{\text{sh}} (\sigma | (\langle w_i \rangle, \langle w_i \rangle), \beta, A)$$

- **LEFT-ARC<sub>k</sub>**,  $k \geq 1$ , or **la<sub>k</sub>** for short. Let  $h$  be the  $k$ -th node in the left spine of the topmost tree in the stack, and let  $d$  be the root node of the second topmost tree in the stack. This transition creates a new arc  $h \rightarrow d$ . Furthermore, the two topmost stack elements are replaced by a new element associated with the tree resulting from the  $h \rightarrow d$  attachment. The transition does not advance with the reading

of the buffer. More formally

$$(\sigma' | \sigma_2 | \sigma_1, \beta, A) \vdash_{\text{la}_k} (\sigma' | \sigma_{\text{la}}, \beta, A \cup \{h \rightarrow d\})$$

where

$$\begin{aligned} \sigma_1 &= (\langle u_{1,1}, \dots, u_{1,p} \rangle, \langle v_{1,1}, \dots, v_{1,q} \rangle), \\ \sigma_2 &= (\langle u_{2,1}, \dots, u_{2,r} \rangle, \langle v_{2,1}, \dots, v_{2,s} \rangle), \\ \sigma_{\text{la}} &= (\langle u_{1,1}, \dots, u_{1,k}, u_{2,1}, \dots, u_{2,r} \rangle, \\ &\quad \langle v_{1,1}, \dots, v_{1,q} \rangle), \end{aligned}$$

and where we have set  $h = u_{1,k}$  and  $d = u_{2,1}$ .

- **RIGHT-ARC<sub>k</sub>**,  $k \geq 1$ , or **ra<sub>k</sub>** for short. This transition is defined symmetrically with respect to **la<sub>k</sub>**. We have

$$(\sigma' | \sigma_2 | \sigma_1, \beta, A) \vdash_{\text{ra}_k} (\sigma' | \sigma_{\text{ra}}, \beta, A \cup \{h \rightarrow d\})$$

where  $\sigma_1$  and  $\sigma_2$  are as in the **la<sub>k</sub>** case,

$$\begin{aligned} \sigma_{\text{ra}} &= (\langle u_{2,1}, \dots, u_{2,r} \rangle, \\ &\quad \langle v_{2,1}, \dots, v_{2,k}, v_{1,1}, \dots, v_{1,q} \rangle), \end{aligned}$$

and we have set  $h = v_{2,k}$  and  $d = v_{1,1}$ .

Transitions **la<sub>k</sub>** and **ra<sub>k</sub>** are parametric in  $k$ , where  $k$  is bounded by the length of the input string and not by a fixed constant (but see also the experimental findings in §5). Thus our system uses an unbounded number of transition relations, which has an apparent disadvantage for learning algorithms. We will get back to this problem in §4.3.

A **complete computation** relative to  $w$  is a sequence of configurations  $c_1, c_2, \dots, c_t$ ,  $t \geq 1$ , such that  $c_1$  and  $c_t$  are initial and final configurations, respectively, and for each  $i \in [2, t]$ ,  $c_i$  is produced by the application of some transition to  $c_{i-1}$ . It is not difficult to see that the transition-based parser specified above is sound, meaning that the set of arcs constructed in any complete computation on  $w$  is always a dependency tree for  $w$ . The parser is also complete, meaning that every (projective) dependency tree for  $w$  is constructed by some complete computation on  $w$ . A mathematical proof of this statement is beyond the scope of this paper, and will not be provided here.

### 3.4 Deterministic Parsing Algorithm

The transition-based parser of the previous section is a nondeterministic device, since several transitions can be applied to a given configuration. This might result in several complete computations

---

**Algorithm 1** Parsing Algorithm

---

**Input:** string  $w = w_0 \cdots w_n$ , function  $\text{score}()$ **Output:** dependency tree  $T_w$  $c = (\sigma, \beta, A) \leftarrow ([], [w_0, \dots, w_n], \emptyset)$ **while**  $|\sigma| > 1 \vee |\beta| > 0$  **do**    **while**  $|\sigma| < 2$  **do**        update  $c$  with sh     $p \leftarrow$  length of left spine of  $\sigma_1$      $s \leftarrow$  length of right spine of  $\sigma_2$      $\mathcal{T} \leftarrow \{\text{la}_k \mid k \in [1, p]\} \cup$          $\{\text{ra}_k \mid k \in [1, s]\} \cup \{\text{sh}\}$      $\text{best}T \leftarrow \text{argmax}_{t \in \mathcal{T}} \text{score}(t, c)$     update  $c$  with  $\text{best}T$ **return**  $T_w = (V_w, A)$ 

---

for  $w$ . We present here an algorithm that runs the parser in pseudo-deterministic mode, greedily choosing at each configuration the transition that maximizes some score function. Algorithm 1 takes as input a string  $w$  and a scoring function  $\text{score}()$  defined over parser transitions and parser configurations. The scoring function will be the subject of §4 and is not discussed here. The output of the parser is a dependency tree for  $w$ .

At each iteration the algorithm checks whether there are at least two elements in the stack and, if this is not the case, it shifts elements from the buffer to the stack. Then the algorithm uses the function  $\text{score}()$  to evaluate all transitions that can be applied under the current configuration  $c = (\sigma, \beta, A)$ , and it applies the transition with the highest score, updating the current configuration.

To parse a sentence of length  $n$  (excluding the root token  $w_0$ ) the algorithm applies exactly  $2n + 1$  transitions. In the worst case, each transition application involves  $1 + p + s$  transition evaluations. We therefore conclude that the algorithm always reaches a configuration with an empty buffer and a stack which contains only one element. Then the algorithm stops, returning the dependency tree whose arc set is defined as in the current configuration.

## 4 Model and Training

In this section we introduce the adopted learning algorithm and discuss the model parameters.

### 4.1 Learning Algorithm

We use a linear model for the score function in Algorithm 1, and define  $\text{score}(t, c) = \vec{w} \cdot \phi(t, c)$ . Here  $\vec{w}$  is a weight vector and function  $\phi$  provides

---

**Algorithm 2** Learning Algorithm

---

**Input:** pair  $(w = w_0 \cdots w_n, A_g)$ , vector  $\vec{w}$ **Output:** vector  $\vec{w}$  $c = (\sigma, \beta, A) \leftarrow ([], [w_0, \dots, w_n], \emptyset)$ **while**  $|\sigma| > 1 \vee |\beta| > 0$  **do**    **while**  $|\sigma| < 2$  **do**        update  $c$  with SHIFT     $p \leftarrow$  length of left spine of  $\sigma_1$      $s \leftarrow$  length of right spine of  $\sigma_2$      $\mathcal{T} \leftarrow \{\text{la}_k \mid k \in [1, p]\} \cup$          $\{\text{ra}_k \mid k \in [1, s]\} \cup \{\text{sh}\}$      $\text{best}T \leftarrow \text{argmax}_{t \in \mathcal{T}} \text{score}(t, c)$      $\text{bestCorrect}T \leftarrow$          $\text{argmax}_{t \in \mathcal{T} \wedge \text{isCorrect}(t)} \text{score}(t, c)$     **if**  $\text{best}T \neq \text{bestCorrect}T$  **then**         $\vec{w} \leftarrow \vec{w} - \phi(\text{best}T, c)$          $+ \phi(\text{bestCorrect}T, c)$     update  $c$  with  $\text{bestCorrect}T$ 

---

a feature vector representation for a transition  $t$  applying to a configuration  $c$ . The function  $\phi$  will be discussed at length in §4.3. The vector  $\vec{w}$  is trained using the perceptron algorithm in combination with the averaging method to avoid overfitting; see Freund and Schapire (1999) and Collins and Duffy (2002) for details.

The training data set consists of pairs  $(w, A_g)$ , where  $w$  is a sentence and  $A_g$  is the set of arcs of the gold (desired) dependency tree for  $w$ . At training time, each pair  $(w, A_g)$  is processed using the learning algorithm described as Algorithm 2. The algorithm is based on the notions of correct and incorrect transitions, discussed at length in §4.2.

Algorithm 2 parses  $w$  following Algorithm 1 and using the current  $\vec{w}$ , until the highest score selected transition  $\text{best}T$  is incorrect according to  $A_g$ . When this happens,  $\vec{w}$  is updated by decreasing the weights of the features associated with the incorrect  $\text{best}T$  and by increasing the weights of the features associated with the transition  $\text{bestCorrect}T$  having the highest score among all possible correct transitions. After each update, the learning algorithm resumes parsing from the current configuration by applying  $\text{bestCorrect}T$ , and moves on using the updated weights.

### 4.2 Correct and Incorrect Transitions

Standard transition-based dependency parsers are trained by associating each gold tree with a canonical complete computation. This means that, for each configuration of interest, only one transition

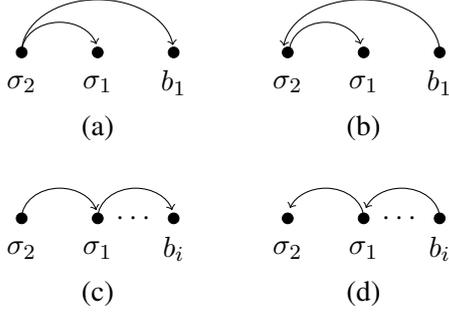


Figure 4: Graphical representation of configurations; drawn arcs are in  $A_g$  but have not yet been added to the configuration. Transition  $sh$  is incorrect for configuration (a) and (b);  $sh$  and  $ra_1$  are correct for (c);  $sh$  and  $la_1$  are correct for (d).

leading to the gold tree is considered as correct. In this paper we depart from such a methodology, and follow Goldberg and Nivre (2012) in allowing more than one correct transition for each configuration, as explained in detail below.

Let  $(w, A_g)$  be a pair in the training set. In §3.3 we have mentioned that there is always a complete computation on  $w$  that results in the construction of the set  $A_g$ . In general, there might be more than one computation for  $A_g$ . This means that the parser shows **spurious ambiguity**.

Observe that all complete computations for  $A_g$  share the same initial configuration  $c_{I,w}$  and final configuration  $c_{F,A_g}$ . Consider now the set  $\mathcal{C}(w)$  of all configurations  $c$  that are reachable from  $c_{I,w}$ , meaning that there exists a sequence of transitions that takes the parser from  $c_{I,w}$  to  $c$ . A configuration  $c \in \mathcal{C}(w)$  is **correct** for  $A_g$  if  $c_{F,A_g}$  is reachable from  $c$ ; otherwise,  $c$  is **incorrect** for  $A_g$ .

Let  $c \in \mathcal{C}(w)$  be a correct configuration for  $A_g$ . A transition  $t$  is correct for  $c$  and  $A_g$  if  $c \vdash_t c'$  and  $c'$  is correct for  $A_g$ ; otherwise,  $t$  is incorrect for  $c$  and  $A_g$ . The next lemma provides a characterization of correct and incorrect transitions; see Figure 4 for examples. We use this characterization in the implementation of predicate  $isCorrect()$  in Algorithm 2.

**Lemma 1** *Let  $(w, A_g)$  be a pair in the training set and let  $c \in \mathcal{C}(w)$  with  $c = (\sigma, \beta, A)$  be a correct configuration for  $A_g$ . Let also  $v_{1,k}$ ,  $k \in [1, q]$ , be the nodes in the right spine of  $\sigma_1$ .*

- (i)  $la_k$  and  $ra_k$  are incorrect for  $c$  and  $A_g$  if and only if they create a new arc  $(h \rightarrow d) \notin A_g$ ;
- (ii)  $sh$  is incorrect for  $c$  and  $A_g$  if and only if the

following conditions are both satisfied:

- (a) there exists an arc  $(h \rightarrow d)$  in  $A_g$  such that  $h$  is in  $\sigma$  and  $d = v_{1,1}$ ;
- (b) there is no arc  $(h' \rightarrow d')$  in  $A_g$  with  $h' = v_{1,k}$ ,  $k \in [1, q]$ , and  $d'$  in  $\beta$ .  $\square$

**PROOF (SKETCH)** To prove part (i) we focus on transition  $ra_k$ ; a similar argument applies to  $la_k$ . The ‘if’ statement in part (i) is self-evident.

‘Only if’. Assuming that transition  $ra_k$  creates a new arc  $(h \rightarrow d) \in A_g$ , we argue that from configuration  $c'$  with  $c \vdash_{ra_k} c'$  we can still reach the final configuration associated with  $A_g$ . We have  $h = v_{2,k}$  and  $d = u_{1,1}$ . The tree fragments in  $\sigma$  with roots  $v_{2,k+1}$  and  $u_{1,1}$  must be adjacent siblings in the tree associated with  $A_g$ , since  $c$  is a correct configuration for  $A_g$  and  $(v_{2,k} \rightarrow u_{1,1}) \in A_g$ . This means that each of the nodes  $v_{2,k+1}, \dots, v_{2,s}$  in the right spine in  $\sigma_2$  in  $c$  must have already acquired all of its right dependents, since the tree is projective. Therefore it is safe for transition  $ra_k$  to eliminate the nodes  $v_{2,k+1}, \dots, v_{2,s}$  from the right spine in  $\sigma_2$ .

We now deal with part (ii). Let  $c \vdash_{sh} c'$ ,  $c' = (\sigma', \beta', A)$ .

‘If’. Assuming (ii)a and (ii)b, we argue that  $c'$  is incorrect. Node  $d$  is the head of  $\sigma'_2$ . Arc  $(h \rightarrow d)$  is not in  $A$ , and the only way we could create  $(h \rightarrow d)$  from  $c'$  is by reaching a new configuration with  $d$  in the topmost stack symbol, which amounts to say that  $\sigma'_1$  can be reduced by a correct transition. Node  $h$  is in some  $\sigma'_i$ ,  $i > 2$ , by (ii)a. Then reduction of  $\sigma'_1$  implies that the root of  $\sigma'_1$  is reachable from the root of  $\sigma'_2$ , which contradicts (ii)b.

‘Only if’. Assuming (ii)a is not satisfied, we argue that  $sh$  is correct for  $c$  and  $A_g$ . There must be an arc  $(h \rightarrow d)$  not in  $A$  with  $d = v_{1,1}$  and  $h$  is some token  $w_i$  in  $\beta$ . From stack  $\sigma' = \sigma''|\sigma'_2|\sigma'_1$  it is always possible to construct  $(h \rightarrow d)$  consuming the substring of  $\beta$  up to  $w_i$  and ending up with stack  $\sigma''|\sigma_{red}$ , where  $\sigma_{red}$  is a stack element with root  $w_i$ . From there, the parser can move on to the final configuration  $c_{F,A_g}$ . A similar argument applies if we assume that (ii)b is not satisfied.  $\blacksquare$

From condition (i) in Lemma 1 and from the fact that there are no cycles in  $A_g$ , it follows that there is at most one correct transition among the transitions of type  $la_k$  or  $ra_k$ . From condition (ii) in the lemma we can also see that the existence of a correct transition of type  $la_k$  or  $ra_k$  for some configuration does not imply that the  $sh$  transition is incorrect

for the same configuration; see Figures 4(c,d) for examples. It follows that for a correct configuration there might be at most 2 correct transitions. In our training experiments for English in §5 we observe 2 correct transitions for 42% of the reached configurations. This nondeterminism is a byproduct of the adopted dynamic parsing strategy, and eventually leads to the spurious ambiguity of the parser.

As already mentioned, we do not impose any canonical form on complete computations that would hardwire a preference for some correct transition and get rid of spurious ambiguity. Following Goldberg and Nivre (2012), we instead regard spurious ambiguity as an additional resource of our parsing strategy. Our main goal is that the training algorithm learns to prefer a *sh* transition in a configuration that does not provide enough information for the choice of the correct arc. In the context of dependency parsing, the strategy of delaying arc construction when the current configuration is not informative is called the *easy-first* strategy, and has been first explored by Goldberg and Elhadad (2010).

### 4.3 Feature Extraction

In existing transition-based parsers a set of atomic features is statically defined and extracted from each configuration. These features are then combined together into complex features, according to some feature template, and joined with the available transition types. This is not possible in our system, since the number of transitions  $la_k$  and  $ra_k$  is not bounded by a constant. Furthermore, it is not meaningful to associate transitions  $la_k$  and  $ra_k$ , for any  $k \geq 1$ , always with the same features, since the constructed arcs impinge on nodes at different depths in the involved spines. It seems indeed more significant to extract information that is local to the arc  $h \rightarrow d$  being constructed by each transition, such as for instance the grandparent and the great grandparent nodes of  $d$ . This is possible if we introduce a higher level of abstraction than in existing transition-based parsers. We remark here that this abstraction also makes the feature representation more similar to the ones typically found in graph-based parsers, which are centered on arcs or subgraphs of the dependency tree.

We index the nodes in the stack  $\sigma$  relative to the head node of the arc being constructed, in case of the transitions  $la_k$  or  $ra_k$ , or else relative to the root node of  $\sigma_1$ , in case of the transition

*sh*. More precisely, let  $c = (\sigma, \beta, A)$  be a configuration and let  $t$  be a transition. We define the **context** of  $c$  and  $t$  as the tuple  $C(c, t) = (s_3, s_2, s_1, q_1, q_2, gp, gg)$ , whose components are placeholders for word tokens in  $\sigma$  or in  $\beta$ . All these placeholders are specified in Table 1, for each  $c$  and  $t$ . Figure 5 shows an example of feature extraction for the displayed configuration  $c = (\sigma, \beta, A)$  and the transition  $la_2$ . In this case we have  $s_3 = u_{3,1}$ ,  $s_2 = u_{2,1}$ ,  $s_1 = u_{1,2}$ ,  $q_1 = gp = u_{1,1}$ ,  $q_2 = b_1$ ;  $gp = none$  because the head of  $gp$  is not available in  $c$ .

Note that in Table 1 placeholders are dynamically assigned in such a way that  $s_1$  and  $s_2$  refer to the nodes in the constructed arc  $h \rightarrow d$ , and  $gp, gg$  refer to the grandparent and the great grandparent nodes, respectively, of  $d$ . Furthermore, the node assigned to  $s_3$  is the parent node of  $s_2$ , if such a node is defined; otherwise, the node assigned to  $s_3$  is the root of the tree fragment in the stack underneath  $\sigma_2$ . Symmetrically, placeholders  $q_1$  and  $q_2$  refer to the parent and grandparent nodes of  $s_1$ , respectively, when these nodes are defined; otherwise, these placeholders get assigned tokens from the buffer. See again Figure 5.

Finally, from the placeholders in  $C(c, t)$  we extract a standard set of atomic features and their complex combinations, to define the function  $\phi$ . Our feature template is an extended version of the feature template of Zhang and Nivre (2011), originally developed for the arc-eager model. The extension is obtained by adding top-down features for left-arcs (based on placeholders  $gp$  and  $gg$ ), and by adding right child features for the first stack element. The latter group of features is usually exploited for the arc-standard model, but is undefined for the arc-eager model.

## 5 Experimental Assessment

Performance evaluation is carried out on the Penn Treebank (Marcus et al., 1993) converted to Stanford basic dependencies (De Marneffe et al., 2006). We use sections 2-21 for training, 22 as development set, and 23 as test set. The part-of-speech tags are assigned by an automatic tagger with accuracy 97.1%. The tagger used on the training set is trained on the same data set by using four-way jackknifing, while the tagger used on the development and test sets is trained on all the training set. We train an arc-labeled version of our parser.

In the first three lines of Table 2 we compare

context placeholder	sh	la <sub>k</sub>			ra <sub>k</sub>		
		k = 1	k = 2	k > 2	k = 1	k = 2	k > 2
s <sub>1</sub>	u <sub>1,1</sub> = v <sub>1,1</sub>	u <sub>1,k</sub>			u <sub>1,1</sub> = v <sub>1,1</sub>		
s <sub>2</sub>	u <sub>2,1</sub> = v <sub>2,1</sub>	u <sub>2,1</sub> = v <sub>2,1</sub>			v <sub>2,k</sub>		
s <sub>3</sub>	u <sub>3,1</sub> = v <sub>3,1</sub>	u <sub>3,1</sub> = v <sub>3,1</sub>			u <sub>3,1</sub> = v <sub>3,1</sub>	v <sub>2,k-1</sub>	
q <sub>1</sub>	b <sub>1</sub>	b <sub>1</sub>	u <sub>1,k-1</sub>		b <sub>1</sub>		
q <sub>2</sub>	b <sub>2</sub>	b <sub>2</sub>	b <sub>2</sub>	u <sub>1,k-2</sub>	b <sub>2</sub>		
gp	none	none	u <sub>1,k-1</sub>		none	v <sub>2,k-1</sub>	
gg	none	none	none	u <sub>1,k-2</sub>	none	none	v <sub>2,k-2</sub>

Table 1: Definition of  $C(c, t) = (s_3, s_2, s_1, q_1, q_2, gp, gg)$ , for  $c = (\sigma'|\sigma_3|\sigma_2|\sigma_1, b_1|b_2|\beta, A)$  and  $t$  of type sh or la<sub>k</sub>, ra<sub>k</sub>,  $k \geq 1$ . Symbols  $u_{j,k}$  and  $v_{j,k}$  are the  $k$ -th nodes in the left and right spines, respectively, of stack element  $\sigma_j$ , with  $u_{j,1} = v_{j,1}$  being the shared root of  $\sigma_j$ ; none is an artificial element used when some context’s placeholder is not available.

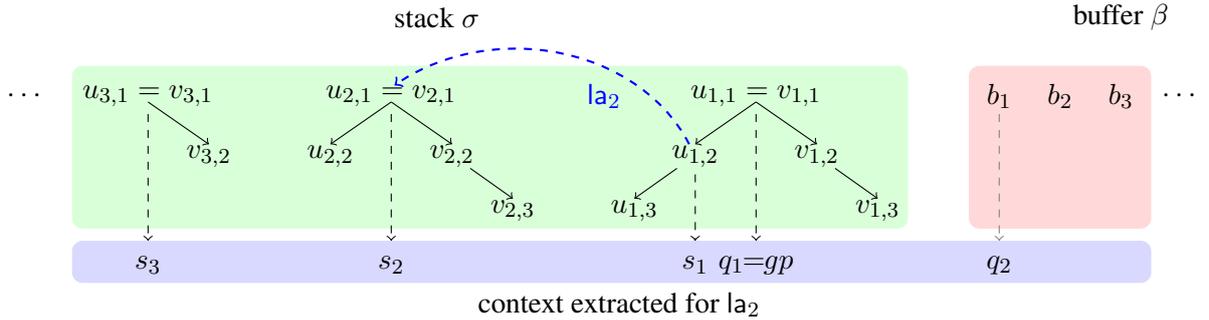


Figure 5: Extraction of atomic features for context  $C(c, la_2) = (s_3, s_2, s_1, q_1, q_2, gp, gg)$ ,  $c = (\sigma, \beta, A)$ .

parser	iter	UAS	LAS	UEM
arc-standard	23	90.02	87.69	38.33
arc-eager	12	90.18	87.83	40.02
this work	30	91.33	89.16	42.38
arc-standard + easy-first	21	90.49	88.22	39.61
arc-standard + spine	27	90.44	88.23	40.27

Table 2: Accuracy on test set, excluding punctuation, for unlabeled attachment score (UAS), labeled attachment score (LAS), unlabeled exact match (UEM).

the accuracy of our parser against our implementation of the arc-eager and arc-standard parsers. For the arc-eager parser, we use the feature template of Zhang and Nivre (2011). The same template is adapted to the arc-standard parser, by removing the top-down parent features and by adding the right child features for the first stack element. It turns out that our feature template, described in §4.3, is the exact merge of the templates used for the arc-eager and the arc-standard parsers.

We train all parsers up to 30 iterations, and for each parser we select the weight vector  $\vec{w}$  from the iteration with the best accuracy on the development set. All our parsers attach the root node at the end of the parsing process, following the ‘None’ ap-

proach discussed by Ballesteros and Nivre (2013). Punctuation is excluded in all evaluation metrics. Considering UAS, our parser provides an improvement of 1.15 over the arc-eager parser and an improvement of 1.31 over the arc-standard parser, that is an error reduction of  $\sim 12\%$  and  $\sim 13\%$ , respectively. Considering LAS, we achieve improvements of 1.33 and 1.47, with an error reduction of  $\sim 11\%$  and  $\sim 12\%$ , over the arc-eager and the arc-standard parsers, respectively.

We speculate that the observed improvement of our parser can be ascribed to two distinct components. The first component is the left-/right-spine representation for stack elements, introduced in §3.3. The second component is the easy-first strategy, implemented on the basis of the spurious ambiguity of our parser and the definition of correct/incorrect transitions in §4.2. In this perspective, we observe that our parser can indeed be viewed as an arc-standard model *augmented* with (i) the spine representation, and (ii) the easy-first strategy. More specifically, (i) generalizes the la/ra transitions to the la<sub>k</sub>/ra<sub>k</sub> transitions, introducing a top-down component into the purely bottom-up arc-standard. On the other hand, (ii) drops the limitation of canonical computations for the arc-standard, and leverages

on the spurious ambiguity of the parser to enlarge the search space.

The two components above are mutually independent, meaning that we can individually implement each component on top of an arc-standard model. More precisely, the arc-standard + spine model uses the transitions  $la_k/ra_k$  but retains the definition of canonical computation, defined by applying each  $la_k/ra_k$  transition as soon as possible. On the other hand, the arc-standard + easy-first model retains the original  $la/ra$  transitions but is trained allowing any correct transition at each configuration. In this case the characterization of correct and incorrect configurations in Lemma 1 has been adapted to transitions  $la/ra$ , taking into account the bottom-up constraint.

With the purpose of incremental comparison, we report accuracy results for the two ‘incremental’ models in the last two lines of Table 2. Analyzing these results, and comparing with the plain arc-standard, we see that the spine representation and the easy-first strategy individually improve accuracy. Moreover, their combination into our model (third line of Table 2) works very well, with an overall improvement larger than the sum of the individual contributions.

We now turn to a computational analysis. At each iteration our parser evaluates a number of transitions bounded by  $\gamma + 1$ , with  $\gamma$  the maximum value of the sum of the lengths of the left spine in  $\sigma_1$  and of the right spine in  $\sigma_2$ . Quantity  $\gamma$  is bounded by the length  $n$  of the input sentence. Since the parser applies exactly  $2n + 1$  transitions, worst case running time is  $\mathcal{O}(n^2)$ . We have computed the average value of  $\gamma$  on our English data set, resulting in 2.98 (variance 2.15) for training set, and 2.95 (variance 1.96) for development set. We conclude that, in the expected case, running time is  $\mathcal{O}(n)$ , with a slow down constant which is rather small, in comparison to standard transition-based parsers. Accordingly, when running our parser against our implementation of the arc-eager and arc-standard models, we measured a slow-down of 2.8 and 2.2, respectively. Besides the change in representation, this slow-down is also due to the increase in the number of features in our system. We have also checked the worst case value of  $\gamma$  in our data set. Interestingly, we have seen that for strings of length smaller than 40 this value linearly grows with  $n$ , and for longer strings the growth stops, with a maximum worst case observed value

of 22.

## 6 Concluding Remarks

We have presented a novel transition-based parser using a dynamic parsing strategy, which achieves a  $\sim 12\%$  error reduction in unlabeled attachment score over the static arc-eager strategy and even more over the (equally static) arc-standard strategy, when evaluated on English.

The idea of representing the right spine of a tree within the stack elements of a shift-reduce device is quite old in parsing, predating empirical approaches. It has been mainly exploited to solve the PP-attachment problem, motivated by psycholinguistic models. The same representation is also adopted in applications of discourse parsing, where right spines are usually called right frontiers; see for instance Subba and Di Eugenio (2009). In the context of transition-based dependency parsers, right spines have also been exploited by Kitagawa and Tanaka-Ishii (2010) to decide where to attach the next word from the buffer. In this paper we have generalized their approach by introducing the symmetrical notion of left spine, and by allowing attachment of full trees rather than attachment of a single word.<sup>2</sup>

Since one can regard a spine as a stack in itself, whose elements are tree nodes, our model is reminiscent of the embedded pushdown automata of Schabes and Vijay-Shanker (1990), used to parse tree adjoining grammars (Joshi and Schabes, 1997) and exploiting a stack of stacks. However, by imposing projectivity, we do not use the extra-power of the latter class.

An interesting line of future research is to combine our dynamic parsing strategy with a training method that allows the parser to explore transitions that apply to incorrect configurations, as in Goldberg and Nivre (2012).

## Acknowledgments

We wish to thank Liang Huang and Marco Kuhlmann for discussion related to the ideas reported in this paper, and the anonymous reviewers for their useful suggestions. The second author has been partially supported by MIUR under project PRIN No. 2010LYA9RH\_006.

---

<sup>2</sup>Accuracy comparison of our work with Kitagawa and Tanaka-Ishii (2010) is not meaningful, since these authors have evaluated their system on the same data set but based on gold part-of-speech tags (personal communication).

## References

- Giuseppe Attardi. 2006. Experiments with a multilingual non-projective dependency parser. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pages 166–170.
- Miguel Ballesteros and Joakim Nivre. 2013. Going to the roots of dependency parsing. *Computational Linguistics*, 39(1):5–13.
- Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465.
- Jinho D. Choi and Martha Palmer. 2011. Getting the most out of transition-based dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 687–692.
- Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 263–270, Philadelphia, Pennsylvania.
- Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, volume 6, pages 449–454.
- Yoav Freund and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, December.
- Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 742–750, Los Angeles, USA.
- Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 959–976.
- Aravind K. Joshi and Yves Schabes. 1997. Tree-Adjoining Grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69–123. Springer.
- Kotaro Kitagawa and Kumiko Tanaka-Ishii. 2010. Tree-based deterministic dependency parsing — an application to Nivre’s method —. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL) Short Papers*, pages 189–193.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.
- Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131.
- Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Workshop on Parsing Technologies (IWPT)*, pages 149–160, Nancy, France.
- Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain.
- Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.
- Yves Schabes and K. Vijay-Shanker. 1990. Deterministic left to right parsing of tree adjoining languages. In *Proceedings of the 28th annual meeting of the Association for Computational Linguistics (ACL)*, pages 276–283, Pittsburgh, Pennsylvania.
- Rajen Subba and Barbara Di Eugenio. 2009. An effective discourse parser that uses rich linguistic information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 566–574.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 195–206.
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 562–571.
- Yue Zhang and Joakim Nivre. 2011. Transition-based parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 188–193.

# Generic binarization for parsing and translation

**Matthias Buechse**

Technische Universität Dresden  
matthias.buechse@tu-dresden.de

**Alexander Koller**

University of Potsdam  
koller@ling.uni-potsdam.de

**Heiko Vogler**

Technische Universität Dresden  
heiko.vogler@tu-dresden.de

## Abstract

Binarization of grammars is crucial for improving the complexity and performance of parsing and translation. We present a versatile binarization algorithm that can be tailored to a number of grammar formalisms by simply varying a formal parameter. We apply our algorithm to binarizing tree-to-string transducers used in syntax-based machine translation.

## 1 Introduction

Binarization amounts to transforming a given grammar into an equivalent grammar of rank 2, i.e., with at most two nonterminals on any right-hand side. The ability to binarize grammars is crucial for efficient parsing, because for many grammar formalisms the parsing complexity depends exponentially on the rank of the grammar. It is also critically important for tractable statistical machine translation (SMT). Syntax-based SMT systems (Chiang, 2007; Graehl et al., 2008) typically use some type of *synchronous grammar* describing a binary translation relation between strings and/or trees, such as synchronous context-free grammars (SCFGs) (Lewis and Stearns, 1966; Chiang, 2007), synchronous tree-substitution grammars (Eisner, 2003), synchronous tree-adjointing grammars (Nesson et al., 2006; DeNeefe and Knight, 2009), and tree-to-string transducers (Yamada and Knight, 2001; Graehl et al., 2008). These grammars typically have a large number of rules, many of which have rank greater than two.

The classical approach to binarization, as known from the Chomsky normal form transformation for context-free grammars (CFGs), proceeds rule by rule. It replaces each rule of rank greater than 2 by an equivalent collection of rules of rank 2. All CFGs can be binarized in this

way, which is why their recognition problem is cubic. In the case of linear context-free rewriting systems (LCFRSs, (Weir, 1988)) the rule-by-rule technique also applies to every grammar, as long as an increased fanout is permitted (Rambow and Satta, 1999).

There are also grammar formalisms for which the rule-by-rule technique is not complete. In the case of SCFGs, not every grammar has an equivalent representation of rank 2 in the first place (Aho and Ullman, 1969). Even when such a representation exists, it is not always possible to compute it rule by rule. Nevertheless, the rule-by-rule binarization algorithm of Huang et al. (2009) is very useful in practice.

In this paper, we offer a generic approach for transferring the rule-by-rule binarization technique to new grammar formalisms. At the core of our approach is a binarization algorithm that can be adapted to a new formalism by changing a parameter at runtime. Thus it only needs to be implemented once, and can then be reused for a variety of formalisms. More specifically, our algorithm requires the user to (i) encode the grammar formalism as a subclass of *interpreted regular tree grammars* (IRTGs, (Koller and Kuhlmann, 2011)) and (ii) supply a collection of *b-rules*, which represent equivalence of grammars syntactically. Our algorithm then replaces, in a given grammar, each rule of rank greater than 2 by an equivalent collection of rules of rank 2, if such a collection is licensed by the b-rules. We define completeness of b-rules in a way that ensures that if any equivalent collection of rules of rank 2 exists, the algorithm finds one. As a consequence, the algorithm binarizes every grammar that can be binarized rule by rule. Step (i) is possible for all the grammar formalisms mentioned above. We show Step (ii) for SCFGs and tree-to-string transducers.

We will use SCFGs as our running example throughout the paper. We will also apply the algo-

rithm to tree-to-string transducers (Graehl et al., 2008; Galley et al., 2004), which describe relations between strings in one language and parse trees of another, which means that existing methods for binarizing SCFGs and LCFRSs cannot be directly applied to these systems. To our knowledge, our binarization algorithm is the first to binarize such transducers. We illustrate the effectiveness of our system by binarizing a large tree-to-string transducer for English-German SMT.

**Plan of the paper.** We start by defining IRTGs in Section 2. In Section 3, we define the general outline of our approach to rule-by-rule binarization for IRTGs, and then extend this to an efficient binarization algorithm based on b-rules in Section 4. In Section 5 we show how to use the algorithm to perform rule-by-rule binarization of SCFGs and tree-to-string transducers, and relate the results to existing work.

## 2 Interpreted regular tree grammars

Grammar formalisms employed in parsing and SMT, such as those mentioned in the introduction, differ in the the derived objects—e.g., strings, trees, and graphs—and the operations involved in the derivation—e.g., concatenation, substitution, and adjoining. Interpreted regular tree grammars (IRTGs) permit a uniform treatment of many of these formalisms. To this end, IRTGs combine two ideas, which we explain here.

**Algebras** IRTGs represent the objects and operations symbolically using terms; the object in question is obtained by interpreting each symbol in the term as a function. As an example, Table 1 shows terms for a string and a tree, together with the denoted object. In the string case, we describe complex strings as concatenation ( $\text{con}^2$ ) of elementary symbols (e.g.,  $a, b$ ); in the tree case, we alternate the construction of a sequence of trees ( $\text{con}^2$ ) with the construction of a single tree by placing a symbol (e.g.,  $\alpha, \beta, \sigma$ ) on top of a (possibly empty) sequence of trees. Whenever a term contains variables, it does not denote an object, but rather a function. In the parlance of universal-algebra theory, we are employing *initial-algebra semantics* (Goguen et al., 1977).

An *alphabet* is a nonempty finite set. Throughout this paper, let  $X = \{x_1, x_2, \dots\}$  be a set, whose elements we call *variables*. We let  $X_k$  denote the set  $\{x_1, \dots, x_k\}$  for every  $k \geq 0$ . Let  $\Sigma$

be an alphabet and  $V \subseteq X$ . We write  $T_\Sigma(V)$  for the set of all terms over  $\Sigma$  with variables  $V$ , i.e., the smallest set  $T$  such that (i)  $V \subseteq T$  and (ii) for every  $\sigma \in \Sigma$ ,  $k \geq 0$ , and  $t_1, \dots, t_k \in T$ , we have  $\sigma(t_1, \dots, t_k) \in T$ . Alternatively, we view  $T_\Sigma(V)$  as the set of all (rooted, labeled, ordered, unranked) *trees over  $\Sigma$  and  $V$* , and draw them as usual. By  $T_\Sigma$  we abbreviate  $T_\Sigma(\emptyset)$ . The *set  $C_\Sigma(V)$  of contexts over  $\Sigma$  and  $V$*  is the set of all trees over  $\Sigma$  and  $V$  in which each variable in  $V$  occurs exactly once.

A *signature* is an alphabet  $\Sigma$  where each symbol is equipped with an *arity*. We write  $\Sigma|_k$  for the subset of all  $k$ -ary symbols of  $\Sigma$ , and  $\sigma|_k$  to denote  $\sigma \in \Sigma|_k$ . We denote the signature by  $\Sigma$  as well. A signature is *binary* if the arities do not exceed 2. Whenever we use  $T_\Sigma(V)$  with a signature  $\Sigma$ , we assume that the trees are ranked, i.e., each node labeled by  $\sigma \in \Sigma|_k$  has exactly  $k$  children.

Let  $\Delta$  be a signature. A  $\Delta$ -*algebra*  $\mathcal{A}$  consists of a nonempty set  $A$  called the *domain* and, for each symbol  $f \in \Delta$  with rank  $k$ , a total function  $f^{\mathcal{A}}: A^k \rightarrow A$ , the *operation* associated with  $f$ . We can *evaluate* any term  $t$  in  $T_\Delta(X_k)$  in  $\mathcal{A}$ , to obtain a  $k$ -ary operation  $t^{\mathcal{A}}$  over the domain. In particular, terms in  $T_\Delta$  evaluate to elements of  $A$ . For instance, in the string algebra shown in Table 1, the term  $\text{con}^2(a, b)$  evaluates to  $ab$ , and the term  $\text{con}^2(\text{con}^2(x_2, a), x_1)$  evaluates to a binary operation  $f$  such that, e.g.,  $f(b, c) = cab$ .

**Bimorphisms** IRTGs separate the finite control (state behavior) of a derivation from its derived object (in its term representation; generational behavior); the former is captured by a regular tree language, while the latter is obtained by applying a tree homomorphism. This idea goes back to the *tree bimorphisms* of Arnold and Dauchet (1976).

Let  $\Sigma$  be a signature. A *regular tree grammar* (RTG)  $G$  over  $\Sigma$  is a triple  $(Q, q_0, R)$  where  $Q$  is a finite set (of *states*),  $q_0 \in Q$ , and  $R$  is a finite set of rules of the form  $q \rightarrow \alpha(q_1, \dots, q_k)$ , where  $q \in Q$ ,  $\alpha \in \Sigma|_k$  and  $q, q_1, \dots, q_k \in Q$ . We call  $\alpha$  the *terminal symbol* and  $k$  the *rank* of the rule. Rules of rank greater than two are called *suprabinary*. For every  $q \in Q$  we define the *language  $L^q(G)$  derived from  $q$*  as the set  $\{\alpha(t_1, \dots, t_k) \mid q \rightarrow \alpha(q_1, \dots, q_k) \in R, t_j \in L^{q_j}(G)\}$ . If  $q = q_0$ , we drop the superscript and write  $L(G)$  for the *tree language of  $G$* . In the literature, there is a definition of RTG which also permits more than one terminal symbol per rule,

	strings over $\Gamma$	trees over $\Gamma$
example term and denoted object	$\text{con}^2 \begin{array}{c} \diagup \\ a \end{array} \begin{array}{c} \diagdown \\ b \end{array} \mapsto ab$	$\begin{array}{c} \sigma \\   \\ \text{con}^2 \\ \diagup \quad \diagdown \\ \alpha \quad \beta \\   \quad   \\ \text{con}^0 \quad \text{con}^0 \end{array} \mapsto \begin{array}{c} \sigma \\ \diagup \quad \diagdown \\ \alpha \quad \beta \end{array}$
domain	$\Gamma^*$	$T_\Gamma^*$ (set of sequences of trees)
signature $\Delta$	$\{a _0 \mid a \in \Gamma\} \cup \{\text{con}^k _k \mid 0 \leq k \leq K, k \neq 1\}$	$\{\gamma _1 \mid \gamma \in \Gamma\} \cup \{\text{con}^k _k \mid 0 \leq k \leq K, k \neq 1\}$
operations	$a: () \mapsto a$ $\text{con}^k: (x_1, \dots, x_k) \mapsto x_1 \cdots x_k$	$\gamma: x_1 \mapsto \gamma(x_1)$ $\text{con}^k: (x_1, \dots, x_k) \mapsto x_1 \cdots x_k$

Table 1: Algebras for strings and trees, given an alphabet  $\Gamma$  and a maximum arity  $K \in \mathbb{N}$ .

or none. This does not increase the generative capacity (Brainerd, 1969).

A (*linear, nondeleting*) *tree homomorphism* is a mapping  $h: T_\Sigma(X) \rightarrow T_\Delta(X)$  that satisfies the following condition: there is a mapping  $g: \Sigma \rightarrow T_\Delta(X)$  such that (i)  $g(\sigma) \in C_\Delta(X_k)$  for every  $\sigma \in \Sigma|_k$ , (ii)  $h(\sigma(t_1, \dots, t_k))$  is the tree obtained from  $g(\sigma)$  by replacing the occurrence of  $x_j$  by  $h(t_j)$ , and (iii)  $h(x_j) = x_j$ . This extends the usual definition of linear and nondeleting homomorphisms (Gécseg and Steinby, 1997) to trees with variables. We abuse notation and write  $h(\sigma)$  for  $g(\sigma)$  for every  $\sigma \in \Sigma$ .

Let  $n \geq 1$  and  $\Delta_1, \dots, \Delta_n$  be signatures. A (*generalized*) *bimorphism* over  $(\Delta_1, \dots, \Delta_n)$  is a tuple  $\mathcal{B} = (G, h_1, \dots, h_n)$  where  $G$  is an RTG over some signature  $\Sigma$  and  $h_i$  is a tree homomorphism from  $T_\Sigma(X)$  into  $T_{\Delta_i}(X)$ . The *language*  $L(\mathcal{B})$  induced by  $\mathcal{B}$  is the tree relation  $\{(h_1(t), \dots, h_n(t)) \mid t \in L(G)\}$ .

An IRTG is a bimorphism whose derived trees are viewed as terms over algebras; see Fig. 1. Formally, an *IRTG*  $\mathbb{G}$  over  $(\Delta_1, \dots, \Delta_n)$  is a tuple  $(\mathcal{B}, \mathcal{A}_1, \dots, \mathcal{A}_n)$  such that  $\mathcal{B}$  is a bimorphism over  $(\Delta_1, \dots, \Delta_n)$  and  $\mathcal{A}_i$  is a  $\Delta_i$ -algebra. The *language*  $L(\mathbb{G})$  induced by  $\mathbb{G}$  is the relation  $\{(t_1^{\mathcal{A}_1}, \dots, t_n^{\mathcal{A}_n}) \mid (t_1, \dots, t_n) \in L(\mathcal{B})\}$ . We call the trees in  $L(G)$  *derivation trees* and the terms in  $L(\mathcal{B})$  *semantic terms*. We say that two IRTGs  $\mathbb{G}$  and  $\mathbb{G}'$  are *equivalent* if  $L(\mathbb{G}) = L(\mathbb{G}')$ . IRTGs were first defined in (Koller and Kuhlmann, 2011).

For example, Fig. 2 is an IRTG that encodes a synchronous context-free grammar (SCFG). It contains a bimorphism  $\mathcal{B} = (G, h_1, h_2)$  consisting of an RTG  $G$  with four rules and homomor-

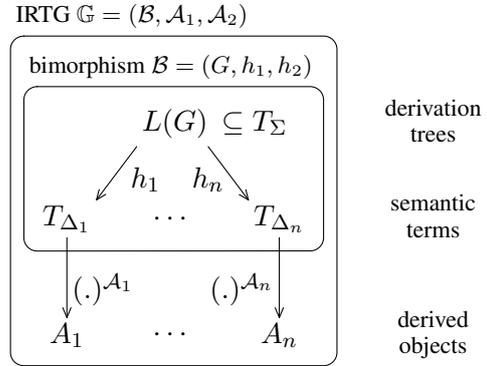


Figure 1: IRTG, bimorphism overview.

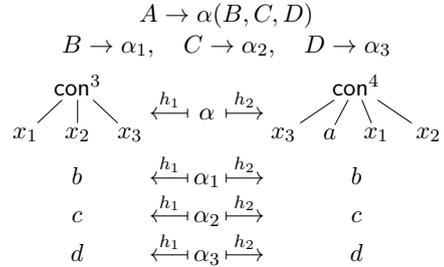


Figure 2: An IRTG encoding an SCFG.

phisms  $h_1$  and  $h_2$  which map derivation trees to trees over the signature of the string algebra in Table 1. By evaluating these trees in the algebra, the symbols  $\text{con}^3$  and  $\text{con}^4$  are interpreted as concatenation, and we see that the first rule encodes the SCFG rule  $A \rightarrow \langle BCD, DaBC \rangle$ . Figure 3 shows a derivation tree with its two homomorphic images, which evaluate to the strings  $bcd$  and  $dabc$ .

IRTGs can be tailored to the expressive capacity of specific grammar formalisms by selecting suitable algebras. The string algebra in Table 1 yields context-free languages, more complex string al-

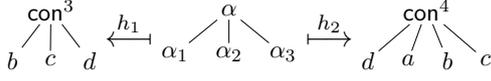


Figure 3: Derivation tree and semantic terms.

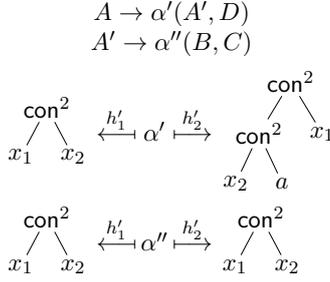


Figure 4: Binary rules corresponding to the  $\alpha$ -rule in Fig. 2.

gebras yield tree-adjointing languages (Koller and Kuhlmann, 2012), and algebras over other domains can yield languages of trees, graphs, or other objects. Furthermore, IRTGs with  $n = 1$  describe languages that are subsets of the algebra's domain,  $n = 2$  yields synchronous languages or tree transductions, and so on.

### 3 IRTG binarization

We will now show how to apply the rule-by-rule binarization technique to IRTGs. We start in this section by defining the binarization of a rule in an IRTG, and characterizing it in terms of *binarization terms* and *variable trees*. We derive the actual binarization algorithm from this in Section 4.

For the remainder of this paper, let  $\mathbb{G} = (\mathcal{B}, \mathcal{A}_1, \dots, \mathcal{A}_n)$  be an IRTG over  $(\Delta_1, \dots, \Delta_n)$  with  $\mathcal{B} = (G, h_1, \dots, h_n)$ .

#### 3.1 An introductory example

We start with an example to give an intuition of our approach. Consider the first rule in Fig. 2, which has rank three. This rule derives (in one step) the fragment  $\alpha(x_1, x_2, x_3)$  of the derivation tree in Fig. 3, which is mapped to the semantic terms  $h_1(\alpha)$  and  $h_2(\alpha)$  shown in Fig. 2. Now consider the rules in Fig. 4. These rules can be used to derive (in two steps) the derivation tree fragment  $\xi$  in Fig. 5e. Note that the terms  $h'_1(\xi)$  and  $h_1(\alpha)$  are *equivalent* in that they denote the same function over the string algebra, and so are the terms  $h'_2(\xi)$  and  $h_2(\alpha)$ . Thus, replacing the  $\alpha$ -rule by the rules in Fig. 4 does not change the language of the IRTG. However, since the new rules are binary,

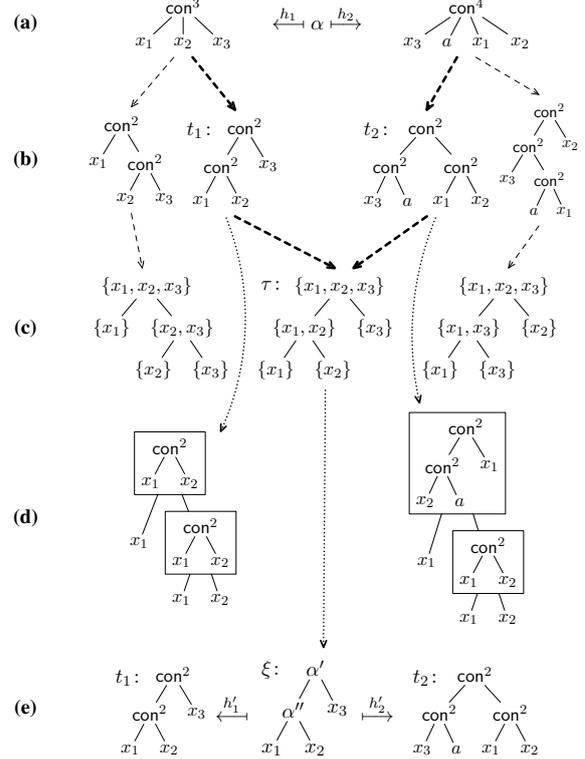


Figure 5: Outline of the binarization algorithm.

parsing and translation will be cheaper.

Now we want to construct the binary rules systematically. In the example, we proceed as follows (cf. Fig. 5). For each of the terms  $h_1(\alpha)$  and  $h_2(\alpha)$  (Fig. 5a), we consider all terms that satisfy two properties (Fig. 5b): (i) they are equivalent to  $h_1(\alpha)$  and  $h_2(\alpha)$ , respectively, and (ii) at each node at most two subtrees contain variables. As Fig. 5 suggests, there may be many different terms of this kind. For each of these terms, we analyze the bracketing of variables, obtaining what we call a *variable tree* (Fig. 5c). Now we pick terms  $t_1$  and  $t_2$  corresponding to  $h_1(\alpha)$  and  $h_2(\alpha)$ , respectively, such that (iii) they have the same variable tree, say  $\tau$ . We construct a tree  $\xi$  from  $\tau$  by a simple relabeling, and we read off the tree homomorphisms  $h'_1$  and  $h'_2$  from a decomposition we perform on  $t_1$  and  $t_2$ , respectively; see Fig. 5, dotted arrows, and compare the boxes in Fig. 5d with the homomorphisms in Fig. 4. Now the rules in Fig. 4 are easily extracted from  $\xi$ .

These rules are equivalent to  $r$  because of (i); they are binary because  $\xi$  is binary, which in turn holds because of (ii); finally, the decompositions of  $t_1$  and  $t_2$  are compatible with  $\xi$  because of (iii). We call terms  $t_1$  and  $t_2$  *binarization terms* if they satisfy (i)–(iii). We will see below that we can con-

struct binary rules equivalent to  $r$  from any given sequence of binarization terms  $t_1, t_2$ , and that binarization terms exist whenever equivalent binary rules exist. The majority of this paper revolves around the question of finding binarization terms.

Rule-by-rule binarization of IRTGs follows the intuition laid out in this example closely: it means processing each suprabinary rule, attempting to replace it with an equivalent collection of binary rules.

### 3.2 Binarization terms

We will now make this intuition precise. To this end, we assume that  $r = q \rightarrow \alpha(q_1, \dots, q_k)$  is a suprabinary rule of  $G$ . As we have seen, binarizing  $r$  boils down to constructing:

- a tree  $\xi$  over some binary signature  $\Sigma'$  and
- tree homomorphisms  $h'_1, \dots, h'_n$  of type  $h'_i: T_{\Sigma'}(X) \rightarrow T_{\Delta_i}(X)$ ,

such that  $h'_i(\xi)$  and  $h_i(\alpha)$  are equivalent, i.e., they denote the same function over  $\mathcal{A}_i$ . We call such a tuple  $(\xi, h'_1, \dots, h'_n)$  a *binarization* of the rule  $r$ . Note that a binarization of  $r$  need not exist. The problem of *rule-by-rule binarization* consists in computing a binarization of each suprabinary rule of a grammar. If such a binarization does not exist, the problem does not have a solution.

In order to define variable trees, we assume a mapping  $\text{seq}$  that maps each finite set  $U$  of pairwise disjoint variable sets to a sequence over  $U$  which contains each element exactly once. Let  $t \in C_{\Delta}(X_k)$ . The *variable set* of  $t$  is the set of all variables that occur in  $t$ . The *set  $S(t)$  of subtree variables* of  $t$  consists of the nonempty variable sets of all subtrees of  $t$ . We represent  $S(t)$  as a tree  $v(t)$ , which we call *variable tree* as follows. Any two elements of  $S(t)$  are either comparable (with respect to the subset relation) or disjoint. We extend this ordering to a tree structure by ordering disjoint elements via  $\text{seq}$ . We let  $v(L) = \{v(t) \mid t \in L\}$  for every  $L \subseteq C_{\Delta}(X_k)$ .

In the example of Fig. 5,  $t_1$  and  $t_2$  have the same set of subtree variables; it is  $\{\{x_1\}, \{x_2\}, \{x_3\}, \{x_1, x_2\}, \{x_1, x_2, x_3\}\}$ . If we assume that  $\text{seq}$  orders sets of variables according to the least variable index, we arrive at the variable tree in the center of Fig. 5.

Now let  $t_1 \in T_{\Delta_1}(X_k), \dots, t_n \in T_{\Delta_n}(X_k)$ . We call the tuple  $t_1, \dots, t_n$  *binarization terms* of  $r$  if the following properties hold: (i)  $h_i(\alpha)$  and  $t_i$  are equivalent; (ii) at each node the tree  $t_i$  contains

at most two subtrees with variables; and (iii) the terms  $t_1, \dots, t_n$  have the same variable tree.

Assume for now that we have found binarization terms  $t_1, \dots, t_n$ . We show how to construct a binarization  $(\xi, h'_1, \dots, h'_n)$  of  $r$  with  $t_i = h'_i(\xi)$ .

First, we construct  $\xi$ . Since  $t_1, \dots, t_n$  are binarization terms, they have the same variable tree, say,  $\tau$ . We obtain  $\xi$  from  $\tau$  by replacing every label of the form  $\{x_j\}$  with  $x_j$ , and every other label with a fresh symbol. Because of condition (ii) in the definition of binarization terms,  $\xi$  is binary.

In order to construct  $h'_i(\sigma)$  for each symbol  $\sigma$  in  $\xi$ , we transform  $t_i$  into a tree  $t'_i$  with labels from  $C_{\Delta_i}(X)$  and the same structure as  $\xi$ . Then we read off  $h'_i(\sigma)$  from the node of  $t'_i$  that corresponds to the  $\sigma$ -labeled node of  $\xi$ . The transformation proceeds as illustrated in Fig. 6: first, we apply the *maximal decomposition operation*  $\rightsquigarrow_d$ ; it replaces every label  $f \in \Delta_i|_k$  by the tree  $f(x_1, \dots, x_k)$ , represented as a box. After that, we keep applying the *merge operation*  $\rightsquigarrow_m$  as often as possible; it merges two boxes that are in a parent-child relation, given that one of them has at most one child. Thus the number of variables in any box can only decrease. Finally, the *reorder operation*  $\rightsquigarrow_o$  orders the children of each box according to the  $\text{seq}$  of their variable sets. These operations do not change the variable tree; one can use this to show that  $t'_i$  has the same structure as  $\xi$ .

Thus, if we can find binarization terms, we can construct a binarization of  $r$ . Conversely, for any given binarization  $(\xi, h'_1, \dots, h'_n)$  the semantic terms  $h'_1(\xi), \dots, h'_n(\xi)$  are binarization terms. This proves the following lemma.

**Lemma 1** *There is a binarization of  $r$  if and only if there are binarization terms of  $r$ .*

### 3.3 Finding binarization terms

It remains to show how we can find binarization terms of  $r$ , if there are any.

Let  $b_i: T_{\Delta_i}(X_k) \rightarrow \mathcal{P}(T_{\Delta_i}(X_k))$  the mapping with  $b_i(t) = \{t' \in T_{\Delta_i}(X_k) \mid t \text{ and } t' \text{ are equivalent, and at each node } t' \text{ has at most two children with variables}\}$ . Figure 5b shows some elements of  $b_1(h_1(\alpha))$  and  $b_2(h_2(\alpha))$  for our example. Terms  $t_1, \dots, t_n$  are binarization terms precisely when  $t_i \in b_i(h_i(\alpha))$  and  $t_1, \dots, t_n$  have the same variable tree. Thus we can characterize binarization terms as follows.

**Lemma 2** *There are binarization terms if and only if  $\bigcap_i v(b_i(h_i(\alpha))) \neq \emptyset$ .*

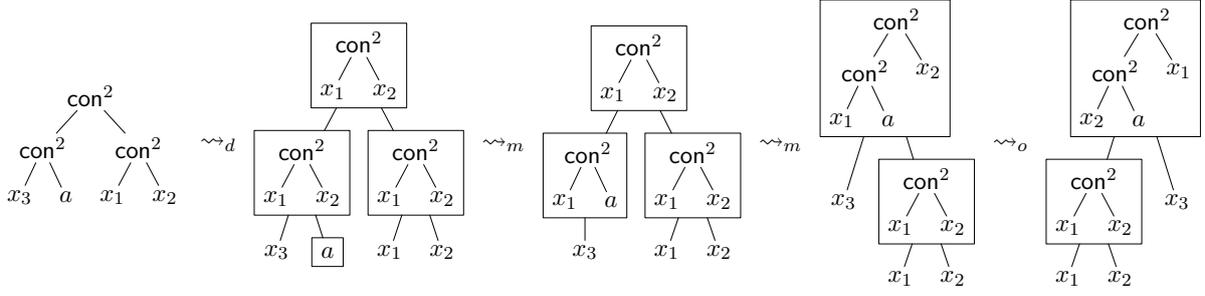


Figure 6: Transforming  $t_2$  into  $t'_2$ .

This result suggests the following procedure for obtaining binarization terms. First, determine whether the intersection in Lemma 2 is empty. If it is, then there is no binarization of  $r$ . Otherwise, select a variable tree  $\tau$  from this set. We know that there are trees  $t_1, \dots, t_n$  such that  $t_i \in b_i(h_i(\alpha))$  and  $v(t_i) = \tau$ . We can therefore select arbitrary concrete trees  $t_i \in b_i(h_i(\alpha)) \cap v^{-1}(\tau)$ . The terms  $t_1, \dots, t_n$  are then binarization terms.

#### 4 Effective IRTG binarization

In this section we develop our binarization algorithm. Its key task is finding binarization terms  $t_1, \dots, t_n$ . This task involves deciding term equivalence, as  $t_i$  must be equivalent to  $h_i(\alpha)$ . In general, equivalence is undecidable, so the task cannot be solved. We avoid deciding equivalence by requiring the user to specify an explicit approximation of  $b_i$ , which we call a *b-rule*. This parameter gives rise to a restricted version of the rule-by-rule binarization problem, which is efficiently computable while remaining practically relevant.

Let  $\Delta$  be a signature. A *binarization rule (b-rule)* over  $\Delta$  is a mapping  $\mathfrak{b}: \Delta \rightarrow \mathcal{P}(T_\Delta(X))$  where for every  $f \in \Delta|_k$  we have that  $\mathfrak{b}(f) \subseteq C_\Delta(X_k)$ , at each node of a tree in  $\mathfrak{b}(f)$  only two children contain variables, and  $\mathfrak{b}(f)$  is a regular tree language. We extend  $\mathfrak{b}$  to  $T_\Delta(X)$  by setting  $\mathfrak{b}(x_j) = \{x_j\}$  and  $\mathfrak{b}(f(t_1, \dots, t_k)) = \{t[x_j/t'_j \mid 1 \leq j \leq k] \mid t \in \mathfrak{b}(f), t'_j \in \mathfrak{b}(t_j)\}$ , where  $[x_j/t'_j]$  denotes substitution of  $x_j$  by  $t'_j$ . Given an algebra  $\mathcal{A}$  over  $\Delta$ , a b-rule  $\mathfrak{b}$  over  $\Delta$  is called a *b-rule over  $\mathcal{A}$*  if, for every  $t \in T_\Delta(X_k)$  and  $t' \in \mathfrak{b}(t)$ ,  $t'$  and  $t$  are equivalent in  $\mathcal{A}$ . Such a b-rule encodes equivalence in  $\mathcal{A}$ , and it does so in an explicit and compact way: because  $\mathfrak{b}(f)$  is a regular tree language, a b-rule can be specified by a finite collection of RTGs, one for each symbol  $f \in \Delta$ . We will look at examples (for the string and tree algebras shown earlier) in Section 5.

From now on, we assume that  $\mathfrak{b}_1, \dots, \mathfrak{b}_n$  are b-rules over  $\mathcal{A}_1, \dots, \mathcal{A}_n$ , respectively. A binarization  $(\xi, h'_1, \dots, h'_n)$  of  $r$  is a *binarization of  $r$  with respect to  $\mathfrak{b}_1, \dots, \mathfrak{b}_n$*  if  $h'_i(\xi) \in \mathfrak{b}_i(h_i(\alpha))$ . Likewise, binarization terms  $t_1, \dots, t_n$  are *binarization terms with respect to  $\mathfrak{b}_1, \dots, \mathfrak{b}_n$*  if  $t_i \in \mathfrak{b}_i(h_i(\alpha))$ . Lemmas 1 and 2 carry over to the restricted notions. The problem of *rule-by-rule binarization with respect to  $\mathfrak{b}_1, \dots, \mathfrak{b}_n$*  consists in computing a binarization with respect to  $\mathfrak{b}_1, \dots, \mathfrak{b}_n$  for each suprabinary rule.

By definition, every solution to this restricted problem is also a solution to the general problem. The converse need not be true. However, we can guarantee that the restricted problem has at least one solution whenever the general problem has one, by requiring  $v(\mathfrak{b}_i(h_i(\alpha))) = v(\mathfrak{b}(h_i(\alpha)))$ . Then the intersection in Lemma 2 is empty in the restricted case if and only if it is empty in the general case. We call the b-rules  $\mathfrak{b}_1, \dots, \mathfrak{b}_1$  *complete* on  $\mathbb{G}$  if the equation holds for every  $\alpha \in \Sigma$ .

Now we show how to effectively compute binarization terms with respect to  $\mathfrak{b}_1, \dots, \mathfrak{b}_n$ , along the lines of Section 3.3. More specifically, we construct an RTG for each of the sets (i)  $\mathfrak{b}_i(h_i(\alpha))$ , (ii)  $\mathfrak{b}'_i = v(\mathfrak{b}_i(h_i(\alpha)))$ , (iii)  $\bigcap_i \mathfrak{b}'_i$ , and (iv)  $\mathfrak{b}''_i = \mathfrak{b}_i(h_i(\alpha)) \cap v^{-1}(\tau)$  (given  $\tau$ ). Then we can select  $\tau$  from (iii) and  $t_i$  from (iv) using a standard algorithm, such as the Viterbi algorithm or Knuth's algorithm (Knuth, 1977; Nederhof, 2003; Huang and Chiang, 2005). The effectiveness of our procedure stems from the fact that we only manipulate RTGs and never enumerate languages.

The construction for (i) is recursive, following the definition of  $\mathfrak{b}_i$ . The base case is a language  $\{x_j\}$ , for which the RTG is easy. For the recursive case, we use the fact that regular tree languages are closed under substitution (Gécseg and Steinby, 1997, Prop. 7.3). Thus we obtain an RTG  $G_i$  with  $L(G_i) = \mathfrak{b}_i(h_i(\alpha))$ .

For (ii) and (iv), we need the following auxiliary

construction. Let  $G_i = (P, p_0, R)$ . We define the mapping  $\text{var}_i: P \rightarrow \mathcal{P}(X_k)$  such that for every  $p \in P$ , every  $t \in L^p(G_i)$  contains exactly the variables in  $\text{var}_i(p)$ . We construct it as follows. We initialize  $\text{var}_i(p)$  to “unknown” for every  $p$ . For every rule  $p \rightarrow x_j$ , we set  $\text{var}_i(p) = \{x_j\}$ . For every rule  $p \rightarrow \sigma(p_1, \dots, p_k)$  such that  $\text{var}_i(p_j)$  is known, we set  $\text{var}_i(p) = \bigcup_j \text{var}_i(p_j)$ . This is iterated; it can be shown that  $\text{var}_i(p)$  is never assigned two different values for the same  $p$ . Finally, we set all remaining unknown entries to  $\emptyset$ .

For (ii), we construct an RTG  $G'_i$  with  $L(G'_i) = b'_i$  as follows. We let  $G'_i = (\{\langle \text{var}_i(p) \rangle \mid p \in P\}, \text{var}_i(p_0), R')$  where  $R'$  consists of the rules

$$\begin{aligned} \langle \{x_j\} \rangle &\rightarrow \{x_j\} && \text{if } p \rightarrow x_i \in R, \\ \langle \text{var}_i(p) \rangle &\rightarrow \text{var}_i(p) \langle \langle U_1 \rangle, \dots, \langle U_l \rangle \rangle \\ &\text{if } p \rightarrow \sigma(p_1, \dots, p_k) \in R, \\ &V = \{\text{var}_i(p_j) \mid 1 \leq j \leq k\} \setminus \{\emptyset\}, \\ &|V| \geq 2, \text{seq}(V) = (U_1, \dots, U_l). \end{aligned}$$

For (iii), we use the standard product construction (Gécseg and Steinby, 1997, Prop. 7.1).

For (iv), we construct an RTG  $G''_i$  such that  $L(G''_i) = b''_i$  as follows. We let  $G''_i = (P, p_0, R'')$ , where  $R''$  consists of the rules

$$\begin{aligned} p &\rightarrow \sigma(p_1, \dots, p_k) \\ &\text{if } p \rightarrow \sigma(p_1, \dots, p_k) \in R, \\ &V = \{\text{var}_i(p_j) \mid 1 \leq j \leq k\} \setminus \{\emptyset\}, \\ &\text{if } |V| \geq 2, \text{ then} \\ &(\text{var}_i(p), \text{seq}(V)) \text{ is a fork in } \tau. \end{aligned}$$

By a fork  $(u, u_1 \dots u_k)$  in  $\tau$ , we mean that there is a node labeled  $u$  with  $k$  children labeled  $u_1$  up to  $u_k$ .

At this point we have all the ingredients for our binarization algorithm, shown in Algorithm 1. It operates directly on a bimorphism, because all the relevant information about the algebras is captured by the b-rules. The following theorem documents the behavior of the algorithm. In short, it solves the problem of rule-by-rule binarization with respect to b-rules  $b_1, \dots, b_n$ .

**Theorem 3** *Let  $\mathbb{G} = (\mathcal{B}, \mathcal{A}_1, \dots, \mathcal{A}_n)$  be an IRTG, and let  $b_1, \dots, b_n$  be b-rules over  $\mathcal{A}_1, \dots, \mathcal{A}_n$ , respectively.*

*Algorithm 1 terminates. Let  $\mathcal{B}'$  be the bimorphism computed by Algorithm 1 on  $\mathcal{B}$  and  $b_1, \dots, b_n$ . Then  $\mathbb{G}' = (\mathcal{B}', \mathcal{A}_1, \dots, \mathcal{A}_n)$  is equivalent to  $\mathbb{G}$ , and  $\mathbb{G}'$  is of rank 2 if and only*

**Input:** bimorphism  $\mathcal{B} = (G, h_1, \dots, h_n)$ ,  
b-rules  $b_1, \dots, b_n$  over  $\Delta_1, \dots, \Delta_n$

**Output:** bimorphism  $\mathcal{B}'$

```

1:  $\mathcal{B}' \leftarrow (G|_{\leq 2}, h_1, \dots, h_n)$ 
2: for rule  $r: q \rightarrow \alpha(q_1, \dots, q_k)$  of  $G|_{>2}$  do
3:   for  $i = 1, \dots, n$  do
4:     compute RTG  $G_i$  for  $b_i(h_i(\alpha))$ 
5:     compute RTG  $G'_i$  for  $v(b_i(h_i(\alpha)))$ 
6:   compute RTG  $G_v$  for  $\bigcap_i L(G'_i)$ 
7:   if  $L(G_v) = \emptyset$  then
8:     add  $r$  to  $\mathcal{B}'$ 
9:   else
10:    select  $t' \in L(G_v)$ 
11:    for  $i = 1, \dots, n$  do
12:      compute RTG  $G''_i$  for
13:         $b''_i = b_i(h_i(\alpha)) \cap v^{-1}(t')$ 
14:      select  $t_i \in L(G''_i)$ 
15:    construct binarization for  $t_1, \dots, t_n$ 
16:    add appropriate rules to  $\mathcal{B}'$ 

```

Algorithm 1: Complete binarization algorithm, where  $G|_{\leq 2}$  and  $G|_{>2}$  is  $G$  restricted to binary and suprabinary rules, respectively.

*if every suprabinary rule of  $\mathbb{G}$  has a binarization with respect to  $b_1, \dots, b_n$ .*

The runtime of Algorithm 1 is dominated by the intersection construction in line 6, which is  $O(m_1 \dots m_n)$  per rule, where  $m_i$  is the size of  $G'_i$ . The quantity  $m_i$  is linear in the size of the terms on the right-hand side of  $h_i$ , and in the number of rules in the b-rule  $b_i$ .

## 5 Applications

Algorithm 1 implements rule-by-rule binarization with respect to given b-rules. If a rule of the given IRTG does not have a binarization with respect to these b-rules, it is simply carried over to the new grammar, which then has a rank higher than 2. The number of remaining suprabinary rules depends on the b-rules (except for rules that have no binarization at all). The user can thus engineer the b-rules according to their current needs, trading off completeness, runtime, and engineering effort.

By contrast, earlier binarization algorithms for formalisms such as SCFG and LCFRS simply attempt to find an equivalent grammar of rank 2; there is no analogue of our b-rules. The problem these algorithms solve corresponds to the general rule-by-rule binarization problem from Section 3.

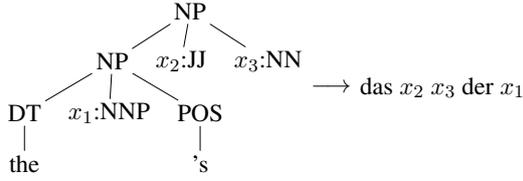


Figure 7: A rule of a tree-to-string transducer.

We show that under certain conditions, our algorithm can be used to solve this problem as well. In the following two subsections, we illustrate this for SCFGs and tree-to-string transducers, respectively. In the final subsection, we discuss how to extend this approach to other grammar formalisms as well.

### 5.1 Synchronous context-free grammars

We have used SCFGs as the running example in this paper. SCFGs are IRTGs with two interpretations into the string algebra of Table 1, as illustrated by the example in Fig. 2. In order to make our algorithm ready to use, it remains to specify a b-rule for the string algebra.

We use the following b-rule for both  $\mathfrak{b}_1$  and  $\mathfrak{b}_2$ . Each symbol  $a \in \Delta_i|_0$  is mapped to the language  $\{a\}$ . Each symbol  $\text{con}^k$ ,  $k \geq 2$ , is mapped to the language induced by the following RTG with states of the form  $[j, j']$  (where  $0 \leq j < j' \leq k$ ) and final state  $[0, k]$ :

$$\begin{aligned}
 [j-1, j] &\rightarrow x_j & (1 \leq j \leq k) \\
 [j, j'] &\rightarrow \text{con}^2([j, j''], [j'', j']) & (0 \leq j < j'' < j' \leq k)
 \end{aligned}$$

This language expresses all possible ways in which  $\text{con}^k$  can be written in terms of  $\text{con}^2$ .

Our definition of rule-by-rule binarization with respect to  $\mathfrak{b}_1$  and  $\mathfrak{b}_2$  coincides with that of Huang et al. (2009): any rule can be binarized by both algorithms or neither. For instance, for the SCFG rule  $A \rightarrow \langle BCDE, CEED \rangle$ , the sets  $v(\mathfrak{b}_1(h_1(\alpha)))$  and  $v(\mathfrak{b}_2(h_2(\alpha)))$  are disjoint, thus no binarization exists. Two strings of length  $N$  can be parsed with a binary IRTG that represents an SCFG in time  $O(N^6)$ .

### 5.2 Tree-to-string transducers

Some approaches to SMT go beyond string-to-string translation models such as SCFG by exploiting known syntactic structures in the source or target language. This perspective on translation naturally leads to the use of tree-to-string transducers

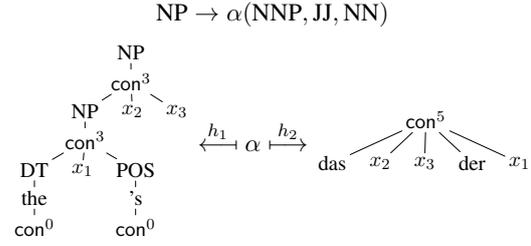


Figure 8: An IRTG rule encoding the rule in Fig. 7.

(Yamada and Knight, 2001; Galley et al., 2004; Huang et al., 2006; Graehl et al., 2008). Figure 7 shows an example of a tree-to-string rule. It might be used to translate “the Commission’s strategic plan” into “das langfristige Programm der Kommission”.

Our algorithm can binarize tree-to-string transducers; to our knowledge, it is the first algorithm to do so. We model the tree-to-string transducer as an IRTG  $\mathbb{G} = ((G, h_1, h_2), \mathcal{A}_1, \mathcal{A}_2)$ , where  $\mathcal{A}_2$  is the string algebra, but this time  $\mathcal{A}_1$  is the tree algebra shown in Table 1. This algebra has operations  $\text{con}^k$  to concatenate sequences of trees and unary  $\gamma$  that maps any sequence  $(t_1, \dots, t_l)$  of trees to the tree  $\gamma(t_1, \dots, t_l)$ , viewed as a sequence of length 1. Note that we exclude the operation  $\text{con}^1$  because it is the identity and thus unnecessary. Thus the rule in Fig. 7 translates to the IRTG rule shown in Fig. 8.

For the string algebra, we reuse the b-rule from Section 5.1; we call it  $\mathfrak{b}_2$  here. For the tree algebra, we use the following b-rule  $\mathfrak{b}_1$ . It maps  $\text{con}^0$  to  $\{\text{con}^0\}$  and each unary symbol  $\gamma$  to  $\{\gamma(x_1)\}$ . Each symbol  $\text{con}^k$ ,  $k \geq 2$ , is treated as in the string case. Using these b-rules, we can binarize the rule in Fig. 8 and obtain the rules in Fig. 9. Parsing of a binary IRTG that represents a tree-to-string transducer is  $O(N^3 \cdot M)$  for a string of length  $N$  and a tree with  $M$  nodes.

We have implemented our binarization algorithm and the b-rules for the string and the tree algebra. In order to test our implementation, we extracted a tree-to-string transducer from about a million parallel sentences of English-German Europarl data, using the GHKM rule extractor (Galley, 2010). Then we binarized the transducer. The results are shown in Fig. 10. Of the 2.15 million rules in the extracted transducer, 460,000 were suprabinary, and 67 % of these could be binarized. Binarization took 4.4 minutes on a single core of an Intel Core i5 2520M processor.

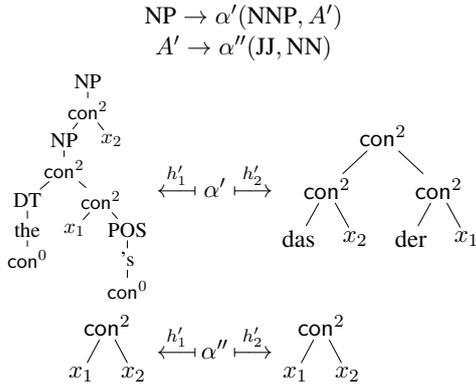


Figure 9: Binarization of the rule in Fig. 8.

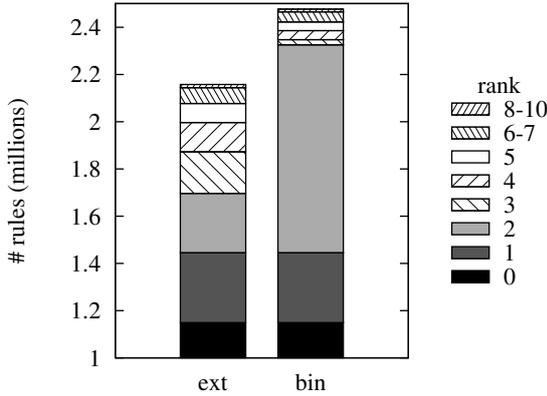


Figure 10: Rules of a transducer extracted from Europarl (ext) vs. its binarization (bin).

### 5.3 General approach

Our binarization algorithm can be used to solve the general rule-by-rule binarization problem for a specific grammar formalism, provided that one can find appropriate b-rules. More precisely, we need to devise a class  $\mathcal{C}$  of IRTGs over the same sequence  $\mathcal{A}_1, \dots, \mathcal{A}_n$  of algebras that encodes the grammar formalism, together with b-rules  $b_1, \dots, b_n$  over  $\mathcal{A}_1, \dots, \mathcal{A}_n$  that are complete on every grammar in  $\mathcal{C}$ , as defined in Section 4.

We have already seen the b-rules for SCFGs and tree-to-string transducers in the preceding subsections; now we have a closer look at the class  $\mathcal{C}$  for SCFGs. We used the class of all IRTGs with two string algebras and in which  $h_i(\alpha)$  contains at most one occurrence of a symbol  $\text{con}^k$  for every  $\alpha \in \Sigma$ . On such a grammar the b-rules are complete. Note that this would not be the case if we allowed several occurrences of  $\text{con}^k$ , as in  $\text{con}^2(\text{con}^2(x_1, x_2), x_3)$ . This term is equivalent to itself and to  $\text{con}^2(x_1, \text{con}^2(x_2, x_3))$ , but the b-

rules only cover the former. Thus they miss one variable tree. For the term  $\text{con}^3(x_1, x_2, x_3)$ , however, the b-rules cover both variable trees.

Generally speaking, given  $\mathcal{C}$  and b-rules  $b_1, \dots, b_n$  that are complete on every IRTG in  $\mathcal{C}$ , Algorithm 1 solves the general rule-by-rule binarization problem on  $\mathcal{C}$ . We can adapt Theorem 3 by requiring that  $\mathbb{G}$  must be in  $\mathcal{C}$ , and replacing each of the two occurrences of “binarization with respect to  $b_1, \dots, b_n$ ” by simply “binarization”. If  $\mathcal{C}$  is such that every grammar from a given grammar formalism can be encoded as an IRTG in  $\mathcal{C}$ , this solves the general rule-by-rule binarization problem of that grammar formalism.

## 6 Conclusion

We have presented an algorithm for binarizing IRTGs rule by rule, with respect to b-rules that the user specifies for each algebra. This improves the complexity of parsing and translation with any monolingual or synchronous grammar that can be represented as an IRTG. A novel algorithm for binarizing tree-to-string transducers falls out as a special case.

In this paper, we have taken the perspective that the binarized IRTG uses the same algebras as the original IRTG. Our algorithm extends to grammars of arbitrary fanout (such as synchronous tree-adjointing grammar (Koller and Kuhlmann, 2012)), but unlike LCFRS-based approaches to binarization, it will not *increase* the fanout to ensure binarizability. In the future, we will explore IRTG binarization with fanout increase. This could be done by binarizing into an IRTG with a more complicated algebra (e.g., of string tuples). We might compute binarizations that are optimal with respect to some measure (e.g., fanout (Gomez-Rodriguez et al., 2009) or parsing complexity (Gildea, 2010)) by keeping track of this measure in the b-rule and taking intersections of weighted tree automata.

## Acknowledgments

We thank the anonymous referees for their insightful remarks, and Sarah Hemmen for implementing an early version of the algorithm. Matthias Büchse was financially supported by DFG VO 1011/6-1.

## References

- Alfred V. Aho and Jeffrey D. Ullman. 1969. Syntax directed translations and the pushdown assembler. *Journal of Computer and System Sciences*, 3:37–56.
- André Arnold and Max Dauchet. 1976. Bistransduction de forêts. In *Proc. 3rd Int. Coll. Automata, Languages and Programming*, pages 74–86. Edinburgh University Press.
- Walter S. Brainerd. 1969. Tree generating regular systems. *Information and Control*, 14(2):217–231.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Steve DeNeefe and Kevin Knight. 2009. Synchronous tree-adjointing machine translation. In *Proceedings of EMNLP*, pages 727–736.
- Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st ACL*, pages 205–208.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a translation rule? In *Proceedings of HLT/NAACL*, pages 273–280.
- Michael Galley. 2010. GHKM rule extractor. <http://www-nlp.stanford.edu/~mgalley/software/stanford-ghkm-latest.tar.gz>, retrieved on March 28, 2012.
- Ferenc Gécseg and Magnus Steinby. 1997. Tree languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer-Verlag.
- Daniel Gildea. 2010. Optimal parsing strategies for linear context-free rewriting systems. In *Proceedings of NAACL HLT*.
- Joseph A. Goguen, Jim W. Thatcher, Eric G. Wagner, and Jesse B. Wright. 1977. Initial algebra semantics and continuous algebras. *Journal of the ACM*, 24:68–95.
- Carlos Gomez-Rodriguez, Marco Kuhlmann, Giorgio Satta, and David Weir. 2009. Optimal reduction of rule length in linear context-free rewriting systems. In *Proceedings of NAACL HLT*.
- Jonathan Graehl, Kevin Knight, and Jonathan May. 2008. Training tree transducers. *Computational Linguistics*, 34(3):391–427.
- Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the 9th IWPT*, pages 53–64.
- Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of the 7th AMTA*, pages 66–73.
- Liang Huang, Hao Zhang, Daniel Gildea, and Kevin Knight. 2009. Binarization of synchronous context-free grammars. *Computational Linguistics*, 35(4):559–595.
- Donald E. Knuth. 1977. A generalization of Dijkstra’s algorithm. *Information Processing Letters*, 6(1):1–5.
- Alexander Koller and Marco Kuhlmann. 2011. A generalized view on parsing and translation. In *Proceedings of the 12th IWPT*, pages 2–13.
- Alexander Koller and Marco Kuhlmann. 2012. Decomposing TAG algorithms using simple algebraizations. In *Proceedings of the 11th TAG+ Workshop*, pages 135–143.
- Philip M. Lewis and Richard E. Stearns. 1966. Syntax directed transduction. *Foundations of Computer Science, IEEE Annual Symposium on*, 0:21–35.
- Mark-Jan Nederhof. 2003. Weighted deductive parsing and Knuth’s algorithm. *Computational Linguistics*, 29(1):135–143.
- Rebecca Nesson, Stuart M. Shieber, and Alexander Rush. 2006. Induction of probabilistic synchronous tree-insertion grammars for machine translation. In *Proceedings of the 7th AMTA*.
- Owen Rambow and Giorgio Satta. 1999. Independent parallelism in finite copying parallel rewriting systems. *Theoretical Computer Science*, 223(1–2):87–120.
- David J. Weir. 1988. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th ACL*, pages 523–530.

# Distortion Model Considering Rich Context for Statistical Machine Translation

Isao Goto<sup>†,‡</sup> Masao Utiyama<sup>†</sup> Eiichiro Sumita<sup>†</sup>  
Akihiro Tamura<sup>†</sup> Sadao Kurohashi<sup>‡</sup>

<sup>†</sup>National Institute of Information and Communications Technology

<sup>‡</sup>Kyoto University

goto.i-es@nhk.or.jp

{mutiyama, eiichiro.sumita, akihiro.tamura}@nict.go.jp

kuro@i.kyoto-u.ac.jp

## Abstract

This paper proposes new distortion models for phrase-based SMT. In decoding, a distortion model estimates the source word position to be translated next (NP) given the last translated source word position (CP). We propose a distortion model that can consider the word at the CP, a word at an NP candidate, and the context of the CP and the NP candidate simultaneously. Moreover, we propose a further improved model that considers richer context by discriminating label sequences that specify spans from the CP to NP candidates. It enables our model to learn the effect of relative word order among NP candidates as well as to learn the effect of distances from the training data. In our experiments, our model improved 2.9 BLEU points for Japanese-English and 2.6 BLEU points for Chinese-English translation compared to the lexical reordering models.

## 1 Introduction

Estimating appropriate word order in a target language is one of the most difficult problems for statistical machine translation (SMT). This is particularly true when translating between languages with widely different word orders.

To address this problem, there has been a lot of research done into word reordering: lexical reordering model (Tillman, 2004), which is one of the distortion models, reordering constraint (Zens et al., 2004), pre-ordering (Xia and McCord, 2004), hierarchical phrase-based SMT (Chiang, 2007), and syntax-based SMT (Yamada and Knight, 2001).

In general, source language syntax is useful for handling long distance word reordering. However,

obtaining syntax requires a syntactic parser, which is not available for many languages. Phrase-based SMT (Koehn et al., 2007) is a widely used SMT method that does not use a parser.

Phrase-based SMT mainly<sup>1</sup> estimates word reordering using distortion models<sup>2</sup>. Therefore, distortion models are one of the most important components for phrase-based SMT. On the other hand, there are methods other than distortion models for improving word reordering for phrase-based SMT, such as pre-ordering or reordering constraints. However, these methods also use distortion models when translating by phrase-based SMT. Therefore, distortion models do not compete against these methods and are commonly used with them. If there is a good distortion model, it will improve the translation quality of phrase-based SMT and benefit to the methods using distortion models.

In this paper, we propose two distortion models for phrase-based SMT. In decoding, a distortion model estimates the source word position to be translated next (NP) given the last translated source word position (CP). The proposed models are *the pair model* and *the sequence model*. The pair model utilizes the word at the CP, a word at an NP candidate site, and the words surrounding the CP and the NP candidates (context) simultaneously. In addition, the sequence model, which is the further improved model, considers richer context by identifying the label sequence that specify the span from the CP to the NP. It enables our model to learn the effect of relative word order among NP candidates as well as to learn the effect of distances from the training data. Our model learns the preference relations among NP

<sup>1</sup>A language model also supports the estimation.

<sup>2</sup>In this paper, reordering models for phrase-based SMT, which are intended to estimate the source word position to be translated next in decoding, are called distortion models. This estimation is used to produce a hypothesis in the target language word order sequentially from left to right.

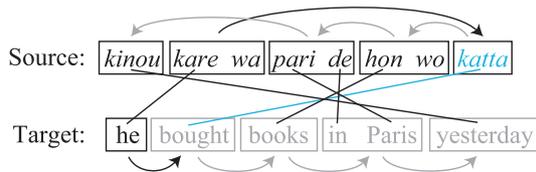


Figure 1: An example of left-to-right translation for Japanese-English. Boxes represent phrases and arrows indicate the translation order of the phrases.

candidates. Our model consists of one probabilistic model and does not require a parser. Experiments confirmed the effectiveness of our method for Japanese-English and Chinese-English translation, using NTCIR-9 Patent Machine Translation Task data sets (Goto et al., 2011).

## 2 Distortion Model for Phrase-Based SMT

A Moses-style phrase-based SMT generates target hypotheses sequentially from left to right. Therefore, the role of the distortion model is to estimate the source phrase position to be translated next whose target side phrase will be located immediately to the right of the already generated hypotheses. An example is shown in Figure 1. In Figure 1, we assume that only the *kare wa* (English side: “he”) has been translated. The target word to be generated next will be “bought” and the source word to be selected next will be its corresponding Japanese word *katta*. Thus, a distortion model should estimate phrases including *katta* as a source phrase position to be translated next.

To explain the distortion model task in more detail, we need to redefine more precisely two terms, the *current position* (CP) and *next position* (NP) in the source sentence. CP is the source sentence position corresponding to the rightmost aligned target word in the generated target word sequence. NP is the source sentence position corresponding to the leftmost aligned target word in the target phrase to be generated next. The task of the distortion model is to estimate the NP<sup>3</sup> from NP candidates (NPCs) for each CP in the source sentence.<sup>4</sup>

<sup>3</sup>NP is not always one position, because there may be multiple correct hypotheses.

<sup>4</sup>This definition is slightly different from that of existing methods such as Moses and (Green et al., 2010). In existing methods, CP is the rightmost position of the last translated source phrase and NP is the leftmost position of the source phrase to be translated next. Note that existing methods do



Figure 2: Examples of CP and NP for Japanese-English translation. The upper sentence is the source sentence and the sentence underneath is a target hypothesis for each example. The NP is in bold, and the CP is in bold italics. The point of an arrow with a  $\times$  mark indicates a wrong NP candidate.

Estimating NP is a difficult task. Figure 2 shows some examples. The superscript numbers indicate the word position in the source sentence.

In Figure 2 (a), the NP is 8. However, in Figure 2 (b), the word (*kare*) at the CP is the same as (a), but the NP is different (the NP is 10). From these examples, we see that distance is not the essential factor in deciding an NP. And it also turns out that the word at the CP alone is not enough to estimate the NP. Thus, not only the word at the CP but also the word at a NP candidate (NPC) should be considered simultaneously.

In (c) and (d) in Figure 2, the word (*kare*) at the CP is the same and *karita* (borrowed) and *katta* (bought) are at the NPCs. *Karita* is the word at the NP and *katta* is not the word at the NP for (c), while *katta* is the word at the NP and *karita* is not the word at the NP for (d). From these examples, considering what the word is at the NP not consider word-level correspondences.

is not enough to estimate the NP. One of the reasons for this difference is the relative word order between words. Thus, considering relative word order is important.

In (d) and (e) in Figure 2, the word (*kare*) at the CP and the word order between *katta* and *karita* are the same. However, the word at the NP for (d) and the word at the NP for (e) are different. From these examples, we can see that selecting a nearby word is not always correct. The difference is caused by the words surrounding the NPCs (context), the CP context, and the words between the CP and the NPC. Thus, these should be considered when estimating the NP.

In summary, in order to estimate the NP, the following should be considered simultaneously: the word at the NP, the word at the CP, the relative word order among the NPCs, the words surrounding NP and CP (context), and the words between the CP and the NPC.

There are distortion models that do not require a parser for phrase-based SMT. The linear distortion cost model used in Moses (Koehn et al., 2007), whose costs are linearly proportional to the reordering distance, always gives a high cost to long distance reordering, even if the reordering is correct. The MSD lexical reordering model (Tillman, 2004; Koehn et al., 2005; Galley and Manning, 2008) only calculates probabilities for the three kinds of phrase reorderings (monotone, swap, and discontinuous), and does not consider relative word order or words between the CP and the NPC. Thus, these models are not sufficient for long distance word reordering.

Al-Onaizan and Papineni (2006) proposed a distortion model that used the word at the CP and the word at an NPC. However, their model did not use context, relative word order, or words between the CP and the NPC.

Ni et al. (2009) proposed a method that adjusts the linear distortion cost using the word at the CP and its context. Their model does not simultaneously consider both the word specified at the CP and the word specified at the NPCs.

Green et al. (2010) proposed distortion models that used context. Their model (the outbound model) estimates how far the NP should be from the CP using the word at the CP and its context.<sup>5</sup> Their model does not simultaneously con-

<sup>5</sup>Their also proposed another model (the inbound model)

sider both the word specified at the CP and the word specified at an NPC. For example, the outbound model considers the word specified at the CP, but does not consider the word specified at an NPC. Their models also do not consider relative word order.

In contrast, our distortion model solves the aforementioned problems. Our distortion models utilize the word specified at the CP, the word specified at an NPC, and also the context of the CP and the NPC simultaneously. Furthermore, our sequence model considers richer context including the relative word order among NPCs and also including all the words between the CP and the NPC. In addition, unlike previous methods, our models learn the preference relations among NPCs.

### 3 Proposed Method

In this section, we first define our distortion model and explain our learning strategy. Then, we describe two proposed models: *the pair model* and *the sequence model* that is the further improved model.

#### 3.1 Distortion Model and Learning Strategy

First, we define our distortion model. Let  $i$  be a CP,  $j$  be an NPC,  $S$  be a source sentence, and  $X$  be the random variable of the NP. In this paper, *distortion probability* is defined as  $P(X = j|i, S)$ , which is the probability of an NPC  $j$  being the NP. Our distortion model is defined as the model calculating the distortion probability.

Next, we explain the learning strategy for our distortion model. We train this model as a discriminative model that discriminates the NP from NPCs. Let  $J$  be a set of word positions in  $S$  other than  $i$ . We train the distortion model subject to

$$\sum_{j \in J} P(X = j|i, S) = 1.$$

The model parameters are learned to maximize the distortion probability of the NP among all of the NPCs  $J$  in each source sentence. This learning strategy is a kind of preference relation learning (Evgeniou and Pontil, 2002). In this learning, the

that estimates reverse direction distance. Each NPC is regarded as an NP, and the inbound model estimates how far the corresponding CP should be from the NP using the word at the NP and its context.

distortion probability of the actual NP will be relatively higher than those of all the other NPCs  $J$ .

This learning strategy is different from that of (Al-Onaizan and Papineni, 2006; Green et al., 2010). For example, Green et al. (2010) trained their outbound model subject to  $\sum_{c \in C} P(Y = c | i, S) = 1$ , where  $C$  is the set of the nine distortion classes<sup>6</sup> and  $Y$  is the random variable of the correct distortion class that the correct distortion is classified into. Distortion is defined as  $j - i - 1$ . Namely, the model probabilities that they learned were the probabilities of distortion classes in all of the training data, not the relative preferences among the NPCs in each source sentence.

### 3.2 Pair Model

The *pair model* utilizes the word at the CP, the word at an NPC, and the context of the CP and the NPC simultaneously to estimate the NP. This can be done by our distortion model definition and the learning strategy described in the previous section.

In this work, we use the maximum entropy method (Berger et al., 1996) as a discriminative machine learning method. The reason for this is that a model based on the maximum entropy method can calculate probabilities. However, if we use scores as an approximation of the distortion probabilities, various discriminative machine learning methods can be applied to build the distortion model.

Let  $s$  be a source word and  $s_1^n = s_1 s_2 \dots s_n$  be a source sentence. We add a beginning of sentence (BOS) marker to the head of the source sentence and an end of sentence (EOS) marker to the end, so the source sentence  $S$  is expressed as  $s_0^{n+1}$  ( $s_0 = \text{BOS}, s_{n+1} = \text{EOS}$ ). Our distortion model calculates the distortion probability for an NPC  $j \in \{j | 1 \leq j \leq n + 1 \wedge j \neq i\}$  for each CP  $i \in \{i | 0 \leq i \leq n\}$

$$P(X = j | i, S) = \frac{1}{Z_i} \exp(\mathbf{w}^T \mathbf{f}(i, j, S, o, d)) \quad (1)$$

where

$$o = \begin{cases} 0 & (i < j) \\ 1 & (i > j) \end{cases}, \quad d = \begin{cases} 0 & (|j - i| = 1) \\ 1 & (2 \leq |j - i| \leq 5) \\ 2 & (6 \leq |j - i|) \end{cases},$$

<sup>6</sup> $(-\infty, -8], [-7, -5], [-4, -3], -2, 0, 1, [2, 3], [4, 6],$  and  $[7, \infty)$ . In (Green et al., 2010),  $-1$  was used as one of distortion classes. However,  $-1$  represents the CP in our definition, and CP is not an NPC. Thus, we shifted all of the distortion classes for negative distortions by  $-1$ .

Template
$\langle o \rangle, \langle o, s_p \rangle^1, \langle o, t_i \rangle, \langle o, t_j \rangle, \langle o, d \rangle, \langle o, s_p, s_q \rangle^2,$
$\langle o, t_i, t_j \rangle, \langle o, t_{i-1}, t_i, t_j \rangle, \langle o, t_i, t_{i+1}, t_j \rangle,$
$\langle o, t_i, t_{j-1}, t_j \rangle, \langle o, t_i, t_j, t_{j+1} \rangle, \langle o, s_i, t_i, t_j \rangle,$
$\langle o, s_j, t_i, t_j \rangle$
<sup>1</sup> $p \in \{p   i - 2 \leq p \leq i + 2 \vee j - 2 \leq p \leq j + 2\}$
<sup>2</sup> $(p, q) \in \{(p, q)   i - 2 \leq p \leq i + 2 \wedge j - 2 \leq q \leq j + 2 \wedge ( p - i  \leq 1 \vee  q - j  \leq 1)\}$

Table 1: Feature templates.  $t$  is the part of speech of  $s$ .

$\mathbf{w}$  is a weight parameter vector, each element of  $\mathbf{f}(\cdot)$  is a binary feature function, and  $Z_i = \sum_{j \in \{j | 1 \leq j \leq n + 1 \wedge j \neq i\}}$  (numerator of Equation 1) is a normalization factor.  $o$  is an orientation of  $i$  to  $j$  and  $d$  is a distance class.

The binary feature function that constitutes an element of  $\mathbf{f}(\cdot)$  returns 1 when its feature is matched and if else, returns 0. Table 1 shows the feature templates used to produce the features. A feature is an instance of a feature template.

In Equation 1,  $i, j$ , and  $S$  are used by the feature functions. Thus, Equation 1 can utilize features consisting of both  $s_i$ , which is the word specified at  $i$ , and  $s_j$ , which is the word specified at  $j$ , or both the context of  $i$  and the context of  $j$  simultaneously. Distance is considered using the distance class  $d$ . Distortion is represented by distance and orientation. The pair model considers distortion using six joint classes of  $d$  and  $o$ .

### 3.3 Sequence Model

The pair model does not consider relative word order among NPCs or all the words between the CP and an NPC. In this section, we propose a further improved model, the *sequence model*, which considers richer context including relative word order among NPCs and also including all the words between the CP and an NPC.

In (c) and (d) in Figure 2, *karita* (borrowed) and *katta* (bought) occur in the source sentences. The pair model considers the effect of distances using only the distance class  $d$ . If these positions are in the same distance class, the pair model cannot consider the differences in distances. In this case, these are conflict instances during training and it is difficult to distinguish the NP for translation.

Now to explain how to consider the relative word order by the sequence model. The sequence model considers the relative word order by discriminating the label sequence corresponding to the NP from the label sequences corresponding to

Label	Description
C	A position is the CP.
I	A position is a position between the CP and the NPC.
N	A position is the NPC.

Table 2: The ‘‘C, I, and N’’ label set.

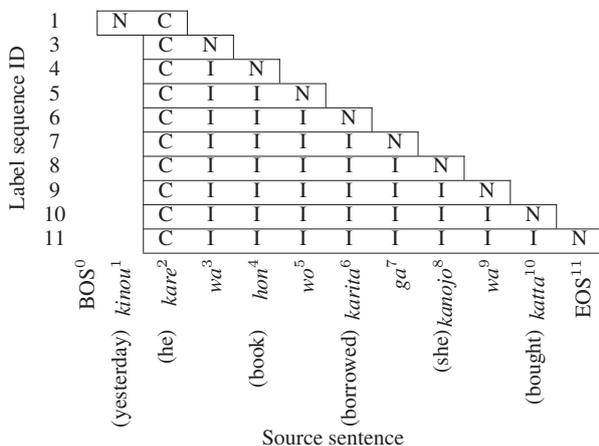


Figure 3: Example of label sequences that specify spans from the CP to each NPC for the case of Figure 2 (c). The labels (C, I, and N) in the boxes are the label sequences.

each NPC in each sentence. Each label sequence corresponds to one NPC. Therefore, if we identify the label sequence that corresponds to the NP, we can obtain the NP. The label sequences specify the spans from the CP to each NPC using three kinds of labels indicating the type of word positions in the spans. The three kinds of labels, ‘‘C, I, and N,’’ are shown in Table 2. Figure 3 shows examples of the label sequences for the case of Figure 2 (c). In Figure 3, the label sequences are represented by boxes and the elements of the sequences are labels. The NPC is used as the label sequence ID for each label sequence.

The label sequence can treat relative word order. For example, the label sequence ID of 10 in Figure 3 knows that *karita* exists to the left of the NPC of 10. This is because *karita*<sup>6</sup> carries a label I while *katta*<sup>10</sup> carries a label N, and a position with label I is defined as relatively closer to the CP than a position with label N. By utilizing the label sequence and corresponding words, the model can reflect the effect of *karita* existing between the CP and the NPC of 10 on the probability.

For the sequence model, *karita* (borrowed) and

*katta* (bought) in (c) and (d) in Figure 2 are not conflict instances in training, whereas they are conflict instances in training for the pair model. The reason is as follows. In order to make the probability of the NPC of 10 smaller than the NPC of 6, instead of making the weight parameters for the features with respect to the word at the position of 10 with label N smaller than the weight parameters for the features with respect to the word at the position of 6 with label N, the sequence model can give negative weight parameters for the features with respect to the word at the position of 6 with label I.

We use a sequence discrimination technique based on CRF (Lafferty et al., 2001) to identify the label sequence that corresponds to the NP. There are two differences between our task and the CRF task. One difference is that CRF discriminates label sequences that consist of labels from all of the label candidates, whereas we constrain the label sequences to sequences where the label at the CP is C, the label at an NPC is N, and the labels between the CP and the NPC are I. The other difference is that CRF is designed for discriminating label sequences corresponding to the same object sequence, whereas we do not assign labels to words outside the spans from the CP to each NPC. However, when we assume that another label such as E has been assigned to the words outside the spans and there are no features involving label E, CRF with our label constraints can be applied to our task. In this paper, the method designed to discriminate label sequences corresponding to the different word sequence lengths is called *partial CRF*.

The sequence model based on partial CRF is derived by extending the pair model. We introduce the label  $l$  and extend the pair model to discriminating the label sequences. There are two extensions to the pair model. One extension uses labels. We suppose that label sequences specify the spans from the CP to each NPC. We conjoined all the feature templates in Table 1 with an additional feature template  $\langle l_i, l_j \rangle$  to include the labels into features where  $l_i$  is the label corresponding to the position of  $i$ . The other extension uses sequence. In the pair model, the position pair of  $(i, j)$  is used to derive features. In contrast, to discriminate label sequences in the sequence model, the position pairs of  $(i, k)$ ,  $k \in \{k | i < k \leq j \vee j \leq k < i\}$

and  $(k, j)$ ,  $k \in \{k | i \leq k < j \vee j < k \leq i\}$  are used to derive features. Note that in the feature templates in Table 1,  $i$  and  $j$  are used to specify two positions. When features are used for the sequence model, one of the positions is regarded as  $k$ .

The distortion probability for an NPC  $j$  being the NP given a CP  $i$  and a source sentence  $S$  is calculated as:

$$P(X = j | i, S) = \frac{1}{Z_i} \exp \left( \sum_{k \in M \cup \{j\}} \mathbf{w}^T \mathbf{f}(i, k, S, o, d, l_i, l_k) + \sum_{k \in M \cup \{i\}} \mathbf{w}^T \mathbf{f}(k, j, S, o, d, l_k, l_j) \right) \quad (2)$$

where

$$M = \begin{cases} \{m | i < m < j\} & (i < j) \\ \{m | j < m < i\} & (i > j) \end{cases}$$

and  $Z_i = \sum_{j \in \{j | 1 \leq j \leq n+1 \wedge j \neq i\}}$  (numerator of Equation 2) is a normalization factor. Since  $j$  is used as the label sequence ID, discriminating  $j$  also means discriminating label sequence IDs.

The first term in  $\exp(\cdot)$  in Equation 2 considers all of the word pairs located at  $i$  and other positions in the sequence, and also their context. The second term in  $\exp(\cdot)$  in Equation 2 considers all of the word pairs located at  $j$  and other positions in the sequence, and also their context.

By designing our model to discriminate among different length label sequences, our model can naturally handle the effect of distances. Many features are derived from a long label sequence because it will contain many labels between the CP and the NPC. On the other hand, fewer features are derived from a short label sequence because a short label sequence will contain fewer labels between the CP and the NPC. The bias from these differences provides important clues for learning the effect of distances.<sup>7</sup>

<sup>7</sup>Note that the sequence model does not only consider larger context than the pair model, but that it also considers labels. The pair model does not discriminate labels, whereas the sequence model uses label N and label I for the positions except for the CP, depending on each situation. For example, in Figure 3, at position 6, label N is used in the label sequence ID of 6, but label I is used in the label sequence IDs of 7 to 11. Namely, even if they are at the same position, the labels in the label sequences are different. The sequence model discriminates the label differences.

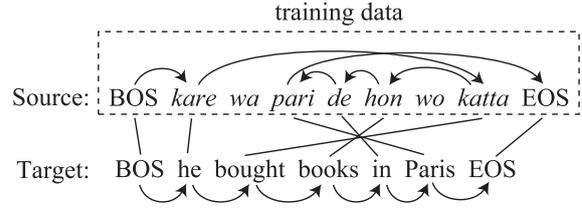


Figure 4: Examples of supervised training data. The lines represent word alignments. The English side arrows point to the nearest word aligned on the right.

### 3.4 Training Data for Discriminative Distortion Model

To train our discriminative distortion model, supervised training data is needed. The training data is built from a parallel corpus and word alignments between corresponding source words and target words. Figure 4 shows examples of training data. We select the target words aligned to the source words sequentially from left to right (target side arrows). Then, the order of the source words in the target word order is decided (source side arrows). The source sentence and the source side arrows are the training data.

## 4 Experiment

In order to confirm the effects of our distortion model, we conducted a series of Japanese to English (JE) and Chinese to English (CE) translation experiments.<sup>8</sup>

### 4.1 Common Settings

We used the patent data for the Japanese to English and Chinese to English translation subtasks from the NTCIR-9 Patent Machine Translation Task (Goto et al., 2011). There were 2,000 sentences for the test data and 2,000 sentences for the development data.

Mecab<sup>9</sup> was used for the Japanese morphological analysis. The Stanford segmenter<sup>10</sup> and tagger<sup>11</sup> were used for Chinese segmentation and POS tagging. The translation model was trained using sentences of 40 words or less from the training data. So approximately 2.05 million sentence pairs consisting of approximately 54 million

<sup>8</sup>We conducted JE and CE translation as examples of language pairs with different word orders and of languages where there is a great need for translation into English.

<sup>9</sup><http://mecab.sourceforge.net/>

<sup>10</sup><http://nlp.stanford.edu/software/segmenter.shtml>

<sup>11</sup><http://nlp.stanford.edu/software/tagger.shtml>

Japanese tokens whose lexicon size was 134k and 50 million English tokens whose lexicon size was 213k were used for JE. And approximately 0.49 million sentence pairs consisting of 14.9 million Chinese tokens whose lexicon size was 169k and 16.3 million English tokens whose lexicon size was 240k were used for CE. GIZA++ and growdiag-final-and heuristics were used to obtain word alignments. In order to reduce word alignment errors, we removed articles {a, an, the} in English and particles {ga, wo, wa} in Japanese before performing word alignments because these function words do not correspond to any words in the other languages. After word alignment, we restored the removed words and shifted the word alignment positions to the original word positions. We used 5-gram language models that were trained using the English side of each set of bilingual training data.

We used an in-house standard phrase-based SMT system compatible with the Moses decoder (Koehn et al., 2007). The SMT weighting parameters were tuned by MERT (Och, 2003) using the development data. To stabilize the MERT results, we tuned three times by MERT using the first half of the development data and we selected the SMT weighting parameter set that performed the best on the second half of the development data based on the BLEU scores from the three SMT weighting parameter sets.

We compared systems that used a common SMT feature set from standard SMT features and different distortion model features. The common SMT feature set consists of: four translation model features, phrase penalty, word penalty, and a language model feature. The compared different distortion model features are: the linear distortion cost model feature (LINEAR), the linear distortion cost model feature and the six MSD bidirectional lexical distortion model (Koehn et al., 2005) features (LINEAR+LEX), the outbound and inbound distortion model features discriminating nine distortion classes (Green et al., 2010) (9-CLASS), the proposed pair model feature (PAIR), and the proposed sequence model feature (SEQUENCE).

## 4.2 Training for the Proposed Models

Our distortion model was trained as follows: We used 0.2 million sentence pairs and their word alignments from the data used to build the translation model as the training data for our distortion models. The features that were selected and used

were the ones that had been counted<sup>12</sup>, using the feature templates in Table 1, at least four times for all of the  $(i, j)$  position pairs in the training sentences. We conjoined the features with three types of label pairs  $\langle C, I \rangle$ ,  $\langle I, N \rangle$ , or  $\langle C, N \rangle$  as instances of the feature template  $\langle l_i, l_j \rangle$  to produce features for SEQUENCE. The L-BFGS method (Liu and Nocedal, 1989) was used to estimate the weight parameters of maximum entropy models. The Gaussian prior (Chen and Rosenfeld, 1999) was used for smoothing.

## 4.3 Training for the Compared Models

For 9-CLASS, we used the same training data as for our distortion models. Let  $t_i$  be the part of speech of  $s_i$ . We used the following feature templates to produce features for the outbound model:  $\langle s_{i-2} \rangle$ ,  $\langle s_{i-1} \rangle$ ,  $\langle s_i \rangle$ ,  $\langle s_{i+1} \rangle$ ,  $\langle s_{i+2} \rangle$ ,  $\langle t_i \rangle$ ,  $\langle t_{i-1}, t_i \rangle$ ,  $\langle t_i, t_{i+1} \rangle$ , and  $\langle s_i, t_i \rangle$ . These feature templates correspond to the components of the feature templates of our distortion models. In addition to these features, we used a feature consisting of the relative source sentence position as the feature used by (Green et al., 2010). The relative source sentence position is discretized into five bins, one for each quintile of the sentence. For the inbound model<sup>13</sup>,  $i$  of the feature templates was changed to  $j$ . Features occurring four or more times in the training sentences were used. The maximum entropy method with Gaussian prior smoothing was used to estimate the model parameters.

The MSD bidirectional lexical distortion model was built using all of the data used to build the translation model.

## 4.4 Results and Discussion

We evaluated translation quality based on the case-insensitive automatic evaluation score BLEU-4 (Papineni et al., 2002). We used distortion limits of 10, 20, 30, and unlimited ( $\infty$ ), which limited the number of words for word reordering to a maximum number. Table 3 presents our main results. The proposed SEQUENCE outperformed the baselines for both Japanese to English and Chinese to English translation. This demonstrates the effectiveness of the proposed SEQUENCE. The scores of the proposed SEQUENCE were higher than those

<sup>12</sup>When we counted features for selection, we only counted features that were from the feature templates of  $\langle s_i, s_j \rangle$ ,  $\langle t_i, t_j \rangle$ ,  $\langle s_i, t_i, t_j \rangle$ , and  $\langle s_j, t_i, t_j \rangle$  in Table 1 when  $j$  was not the NP, in order to avoid increasing the number of features.

<sup>13</sup>The inbound model is explained in footnote 5.

Distortion limit	Japanese-English				Chinese-English			
	10	20	30	$\infty$	10	20	30	$\infty$
LINEAR	27.98	27.74	27.75	27.30	29.18	28.74	28.31	28.33
LINEAR+LEX	30.25	30.37	30.17	29.98	30.81	30.24	30.16	30.13
9-CLASS	30.74	30.98	30.92	30.75	31.80	31.56	31.31	30.84
PAIR	31.62	32.36	31.96	32.03	32.51	32.30	32.25	32.32
SEQUENCE	32.02	32.96	<b>33.29</b>	32.81	<b>33.41</b>	<b>33.44</b>	<b>33.35</b>	<b>33.41</b>

Table 3: Evaluation results for each method. The values are case-insensitive BLEU scores. Bold numbers indicate no significant difference from the best result in each language pair using the bootstrap resampling test at a significance level  $\alpha = 0.01$  (Koehn, 2004).

	Japanese-English	Chinese-English
HIER	30.47	32.66

Table 4: Evaluation results for hierarchical phrase-based SMT.

of the proposed PAIR. This confirms the effectiveness for considering relative word order and words between the CP and an NPC. The proposed PAIR outperformed 9-CLASS, confirming that considering both the word specified at the CP and the word specified at the NPC simultaneously was more effective than that of 9-CLASS.

For translating between languages with widely different word orders such as Japanese and English, a small distortion limit is undesirable because there are cases where correct translations cannot be produced with a small distortion limit, since the distortion limit prunes the search space that does not meet the constraint. Therefore, a large distortion limit is required to translate correctly. For JE translation, our SEQUENCE achieved significantly better results at distortion limits of 20 and 30 than that at a distortion limit of 10, while the baseline systems of LINEAR, LINEAR+LEX, and 9-CLASS did not achieve this. This indicates that SEQUENCE could treat long distance reordering candidates more appropriately than the compared methods.

We also tested hierarchical phrase-based SMT (Chiang, 2007) (HIER) using the Moses implementation. The common data was used to train HIER. We used unlimited max-chart-span for the system setting. Results are given in Table 4. Our SEQUENCE outperformed HIER. The gain for JE was large but the gain for CE was modest. Since phrase-based SMT is generally faster in decoding speed than hierarchical phrase-based SMT, achieving better or comparable scores is worth-

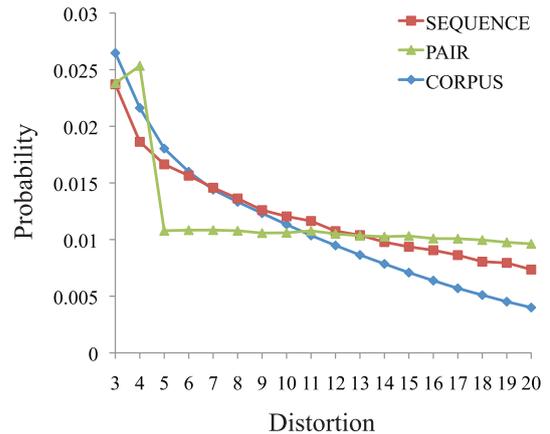


Figure 5: Average probabilities for large distortion for Japanese-English translation.

while.

To investigate the tolerance for sparsity of the training data, we reduced the training data for the sequence model to 20,000 sentences for JE translation.<sup>14</sup> SEQUENCE using this model with a distortion limit of 30 achieved a BLEU score of 32.22.<sup>15</sup> Although the score is lower than the score of SEQUENCE with a distortion limit of 30 in Table 3, the score was still higher than those of LINEAR, LINEAR+LEX, and 9-CLASS for JE in Table 3. This indicates that the sequence model also works even when the training data is not large. This is because the sequence model considers not only the word at the CP and the word at an NPC but also rich context, and rich context would be effective even for a smaller set of training data.

<sup>14</sup>We did not conduct experiments using larger training data because there would have been a very high computational cost to build models using the L-BFGS method.

<sup>15</sup>To avoid effects from differences in the SMT weighting parameters, we used the same SMT weighting parameters for SEQUENCE, with a distortion limit of 30, in Table 3.

To investigate how well SEQUENCE learns the effect of distance, we checked the average distortion probabilities for large distortions of  $j - i - 1$ . Figure 5 shows three kinds of probabilities for distortions from 3 to 20 for Japanese-English translation. One is the average distortion probabilities in the Japanese test sentences for each distortion for SEQUENCE, and another is this for PAIR. The third (CORPUS) is the probabilities for the actual distortions in the training data that were obtained from the word alignments used to build the translation model. The probability for a distortion for CORPUS was calculated by the number of the distortion divided by the total number of distortions in the training data.

Figure 5 shows that when a distance class feature used in the model was the same (e.g., distortions from 5 to 20 were the same distance class feature), PAIR produced average distortion probabilities that were almost the same. In contrast, the average distortion probabilities for SEQUENCE decreased when the lengths of the distortions increased, even if the distance class feature was the same, and this behavior was the same as that of CORPUS. This confirms that the proposed SEQUENCE could learn the effect of distances appropriately from the training data.<sup>16</sup>

## 5 Related Works

We discuss related works other than discussed in Section 2. Xiong et al. (2012) proposed a model predicting the orientation of an argument with respect to its verb using a parser. Syntactic structures and predicate-argument structures are useful for reordering. However, orientations do not handle distances. Thus, our distortion model does not compete against the methods predicting orientations using a parser and would assist them if used

<sup>16</sup>We also checked the average distortion probabilities for the 9-CLASS outbound model in the Japanese test sentences for Japanese-English translation. We averaged the average probabilities for distortions in a distortion span of [4, 6] and also averaged those in a distortion span of [7, 20], where the distortions in each span are in the same distortion class. The average probability for [4, 6] was 0.058 and that for [7, 20] was 0.165. From CORPUS, the average probabilities in the training data for each distortion in [4, 6] were higher than those for each distortion in [7, 20]. However, the converse was true for the comparison between the two average probabilities for the outbound model. This is because the sum of probabilities for distortions from 7 and above was larger than the sum of probabilities for distortions from 4 to 6 in the training data. This comparison indicates that the 9-CLASS outbound model could not appropriately learn the effects of large distances for JE translation.

together.

There are word reordering constraint methods using ITG (Wu, 1997) for phrase-based SMT (Zens et al., 2004; Yamamoto et al., 2008; Feng et al., 2010). These methods consider sentence level consistency with respect to ITG. The ITG constraint does not consider distances of reordering and was used with other distortion models. Our distortion model does not consider sentence level consistency, so our distortion model and ITG constraint methods are thought to be complementary.

There are tree-based SMT methods (Chiang, 2007; Galley et al., 2004; Liu et al., 2006). In many cases, tree-based SMT methods do not use the distortion models that consider reordering distance apart from translation rules because it is not trivial to use distortion scores considering the distances for decoders that do not generate hypotheses from left to right. If it could be applied to these methods, our distortion model might contribute to tree-based SMT methods. Investigating the effects will be for future work.

## 6 Conclusion

This paper described our distortion models for phrase-based SMT. Our sequence model simply consists of only one probabilistic model, but it can consider rich context. Experiments indicate that our models achieved better performance and the sequence model could learn the effect of distances appropriately. Since our models do not require a parser, they can be applied to many languages. Future work includes application to other language pairs, incorporation into ITG constraint methods and other reordering methods, and application to tree-based SMT methods.

## References

- Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 529–536, Sydney, Australia, July. Association for Computational Linguistics.
- Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March.
- Stanley F. Chen and Ronald Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical report.

- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Theodoros Evgeniou and Massimiliano Pontil. 2002. Learning preference relations from data. *Neural Nets Lecture Notes in Computer Science*, 2486:23–32.
- Yang Feng, Haitao Mi, Yang Liu, and Qun Liu. 2010. An efficient shift-reduce decoding algorithm for phrased-based machine translation. In *Coling 2010: Posters*, pages 285–293, Beijing, China, August. Coling 2010 Organizing Committee.
- Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a translation rule? In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the NTCIR-9 workshop. In *Proceedings of NTCIR-9*, pages 559–578.
- Spence Green, Michel Galley, and Christopher D. Manning. 2010. Improved models of distortion cost for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 867–875, Los Angeles, California, June. Association for Computational Linguistics.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of 18th International Conference on Machine Learning*, pages 282–289.
- D.C. Liu and J. Nocedal. 1989. On the limited memory method for large scale optimization. *Mathematical Programming B*, 45(3):503–528.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia, July. Association for Computational Linguistics.
- Yizhao Ni, Craig Saunders, Sandor Szedmak, and Mahesan Niranjan. 2009. Handling phrase reorderings for machine translation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 241–244, Suntec, Singapore, August. Association for Computational Linguistics.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.
- Christoph Tillman. 2004. A unigram orientation model for statistical machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 101–104, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.
- Fei Xia and Michael McCord. 2004. Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of Coling 2004*, pages 508–514, Geneva, Switzerland, Aug 23–Aug 27. COLING.
- Deyi Xiong, Min Zhang, and Haizhou Li. 2012. Modeling the translation of predicate-argument structure for smt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 902–911, Jeju Island, Korea, July. Association for Computational Linguistics.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Toulouse, France, July. Association for Computational Linguistics.

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

Richard Zens, Hermann Ney, Taro Watanabe, and Eiichiro Sumita. 2004. Reordering constraints for phrase-based statistical machine translation. In *Proceedings of Coling 2004*, pages 205–211, Geneva, Switzerland, Aug 23–Aug 27. COLING.

# Word Alignment Modeling with Context Dependent Deep Neural Network

Nan Yang<sup>1</sup>, Shujie Liu<sup>2</sup>, Mu Li<sup>2</sup>, Ming Zhou<sup>2</sup>, Nenghai Yu<sup>1</sup>

<sup>1</sup>University of Science and Technology of China, Hefei, China

<sup>2</sup>Microsoft Research Asia, Beijing, China

{v-nayang, shujliu, muli, mingzhou}@microsoft.com  
ynh@ustc.edu.cn

## Abstract

In this paper, we explore a novel bilingual word alignment approach based on DNN (Deep Neural Network), which has been proven to be very effective in various machine learning tasks (Collobert et al., 2011). We describe in detail how we adapt and extend the CD-DNN-HMM (Dahl et al., 2012) method introduced in speech recognition to the HMM-based word alignment model, in which bilingual word embedding is discriminatively learnt to capture lexical translation information, and surrounding words are leveraged to model context information in bilingual sentences. While being capable to model the rich bilingual correspondence, our method generates a very compact model with much fewer parameters. Experiments on a large scale English-Chinese word alignment task show that the proposed method outperforms the HMM and IBM model 4 baselines by 2 points in F-score.

## 1 Introduction

Recent years research communities have seen a strong resurgent interest in modeling with deep (multi-layer) neural networks. This trending topic, usually referred under the name *Deep Learning*, is started by ground-breaking papers such as (Hinton et al., 2006), in which innovative training procedures of deep structures are proposed. Unlike shallow learning methods, such as Support Vector Machine, Conditional Random Fields, and Maximum Entropy, which need hand-craft features as input, DNN can learn suitable features (representations) automatically with raw input data, given a training objective.

DNN did not achieve expected success until 2006, when researchers discovered a proper way

to initialize and train the deep architectures, which contains two phases: layer-wise unsupervised pre-training and supervised fine tuning. For pre-training, Restricted Boltzmann Machine (RBM) (Hinton et al., 2006), auto-encoding (Bengio et al., 2007) and sparse coding (Lee et al., 2007) are proposed and popularly used. The unsupervised pre-training trains the network one layer at a time, and helps to guide the parameters of the layer towards better regions in parameter space (Bengio, 2009). Followed by fine tuning in this region, DNN is shown to be able to achieve state-of-the-art performance in various area, or even better (Dahl et al., 2012) (Kavukcuoglu et al., 2010). DNN also achieved breakthrough results on the ImageNet dataset for objective recognition (Krizhevsky et al., 2012). For speech recognition, (Dahl et al., 2012) proposed context-dependent neural network with large vocabulary, which achieved 16.0% relative error reduction.

DNN has also been applied in Natural Language Processing (NLP) field. Most works convert atomic lexical entries into a dense, low dimensional, real-valued representation, called *word embedding*; Each dimension represents a latent aspect of a word, capturing its semantic and syntactic properties (Bengio et al., 2006). Word embedding is usually first learned from huge amount of monolingual texts, and then fine-tuned with task-specific objectives. (Collobert et al., 2011) and (Socher et al., 2011) further apply Recursive Neural Networks to address the structural prediction tasks such as tagging and parsing, and (Socher et al., 2012) explores the compositional aspect of word representations.

Inspired by successful previous works, we propose a new DNN-based word alignment method, which exploits contextual and semantic similarities between words. As shown in example (a) of Figure 1, in word pair {“juda”  $\Rightarrow$  “mammoth”}, the Chinese word “juda” is a common word, but

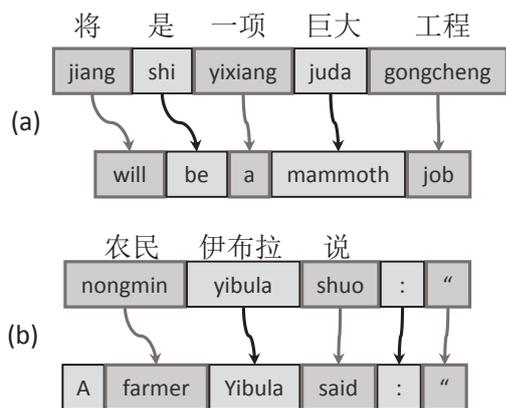


Figure 1: Two examples of word alignment

the English word “mammoth” is not, so it is very hard to align them correctly. If we know that “mammoth” has the similar meaning with “big”, or “huge”, it would be easier to find the corresponding word in the Chinese sentence. As we mentioned in the last paragraph, *word embedding* (trained with huge monolingual texts) has the ability to map a word into a vector space, in which, similar words are near each other.

For example (b) in Figure 1, for the word pair {“yibula”  $\Rightarrow$  “Yibula”}, both the Chinese word “yibula” and English word “Yibula” are rare name entities, but the words around them are very common, which are {“nongmin”, “shuo”} for Chinese side and {“farmer”, “said”} for the English side. The pattern of the context {“nongmin  $X$  shuo”  $\Rightarrow$  “farmer  $X$  said”} may help to align the word pair which fill the variable  $X$ , and also, the pattern {“yixiang  $X$  gongcheng”  $\Rightarrow$  “a  $X$  job”} is helpful to align the word pair {“juda”  $\Rightarrow$  “mammoth”} for example (a).

Based on the above analysis, in this paper, both the words in the source and target sides are firstly mapped to a vector via a discriminatively trained word embeddings, and word pairs are scored by a multi-layer neural network which takes rich contexts (surrounding words on both source and target sides) into consideration; and a HMM-like distortion model is applied on top of the neural network to characterize structural aspect of bilingual sentences.

In the rest of this paper, related work about DNN and word alignment are first reviewed in Section 2, followed by a brief introduction of DNN in Section 3. We then introduce the details of leveraging DNN for word alignment, including the details of our network structure in Section 4

and the training method in Section 5. The merits of our approach are illustrated with the experiments described in Section 6, and we conclude our paper in Section 7.

## 2 Related Work

DNN with unsupervised pre-training was firstly introduced by (Hinton et al., 2006) for MNIST digit image classification problem, in which, RBM was introduced as the layer-wise pre-trainer. The layer-wise pre-training phase found a better local maximum for the multi-layer network, thus led to improved performance. (Krizhevsky et al., 2012) proposed to apply DNN to do object recognition task (ImageNet dataset), which brought down the state-of-the-art error rate from 26.1% to 15.3%. (Seide et al., 2011) and (Dahl et al., 2012) apply Context-Dependent Deep Neural Network with HMM (CD-DNN-HMM) to speech recognition task, which significantly outperforms traditional models.

Most methods using DNN in NLP start with a word embedding phase, which maps words into a fixed length, real valued vectors. (Bengio et al., 2006) proposed to use multi-layer neural network for language modeling task. (Collobert et al., 2011) applied DNN on several NLP tasks, such as part-of-speech tagging, chunking, name entity recognition, semantic labeling and syntactic parsing, where they got similar or even better results than the state-of-the-art on these tasks. (Niehues and Waibel, 2012) shows that machine translation results can be improved by combining neural language model with n-gram traditional language. (Son et al., 2012) improves translation quality of n-gram translation model by using a bilingual neural language model. (Titov et al., 2012) learns a context-free cross-lingual word embeddings to facilitate cross-lingual information retrieval.

For the related works of word alignment, the most popular methods are based on generative models such as IBM Models (Brown et al., 1993) and HMM (Vogel et al., 1996). Discriminative approaches are also proposed to use hand crafted features to improve word alignment. Among them, (Liu et al., 2010) proposed to use phrase and rule pairs to model the context information in a log-linear framework. Unlike previous discriminative methods, in this work, we do not resort to any hand crafted features, but use DNN to induce “features” from raw words.

### 3 DNN structures for NLP

The most important and prevalent features available in NLP are the words themselves. To apply DNN to NLP task, the first step is to transform a discrete word into its *word embedding*, a low dimensional, dense, real-valued vector (Bengio et al., 2006). Word embeddings often implicitly encode syntactic or semantic knowledge of the words. Assuming a finite sized vocabulary  $V$ , word embeddings form a  $(L \times |V|)$ -dimension embedding matrix  $W_V$ , where  $L$  is a pre-determined embedding length; mapping words to embeddings is done by simply looking up their respective columns in the embedding matrix  $W_V$ . The lookup process is called a *lookup layer LT*, which is usually the first layer after the input layer in neural network.

After words have been transformed to their embeddings, they can be fed into subsequent classical network layers to model highly non-linear relations:

$$z^l = f_l(M^l z^{l-1} + b^l) \quad (1)$$

where  $z^l$  is the output of  $l$ th layer,  $M^l$  is a  $|z^l| \times |z^{l-1}|$  matrix,  $b^l$  is a  $|z^l|$ -length vector, and  $f_l$  is an *activation* function. Except for the last layer,  $f_l$  must be non-linear. Common choices for  $f_l$  include sigmoid function, hyperbolic function, “hard” hyperbolic function etc. Following (Collobert et al., 2011), we choose “hard” hyperbolic function as our activation function in this work:

$$htanh(x) = \begin{cases} 1 & \text{if } x \text{ is greater than } 1 \\ -1 & \text{if } x \text{ is less than } -1 \\ x & \text{otherwise} \end{cases} \quad (2)$$

If probabilistic interpretation is desired, a *softmax* layer (Bridle, 1990) can be used to do normalization:

$$z_i^l = \frac{e^{z_i^{l-1}}}{\sum_{j=1}^{|z^l|} e^{z_j^{l-1}}} \quad (3)$$

The above layers can only handle fixed sized input and output. If input must be of variable length, *convolution* layer and *max* layer can be used, (Collobert et al., 2011) which transform variable length input to fixed length vector for further processing.

Multi-layer neural networks are trained with the standard *back propagation* algorithm (LeCun, 1985). As the networks are non-linear and the task specific objectives usually contain many local maximums, special care must be taken in the

optimization process to obtain good parameters. Techniques such as *layerwise pre-training* (Bengio et al., 2007) and many tricks (LeCun et al., 1998) have been developed to train better neural networks. Besides that, neural network training also involves some hyperparameters such as learning rate, the number of hidden layers. We will address these issues in section 4.

### 4 DNN for word alignment

Our DNN word alignment model extends classic HMM word alignment model (Vogel et al., 1996). Given a sentence pair  $(\mathbf{e}, \mathbf{f})$ , HMM word alignment takes the following form:

$$P(\mathbf{a}, \mathbf{e}|\mathbf{f}) = \prod_{i=1}^{|\mathbf{e}|} P_{lex}(e_i|f_{a_i})P_d(a_i - a_{i-1}) \quad (4)$$

where  $P_{lex}$  is the lexical translation probability and  $P_d$  is the jump distance distortion probability.

One straightforward way to integrate DNN into HMM is to use neural network to compute the emission (lexical translation) probability  $P_{lex}$ . Such approach requires a softmax layer in the neural network to normalize over all words in source vocabulary. As vocabulary for natural languages is usually very large, it is prohibitively expensive to do the normalization. Hence we give up the probabilistic interpretation and resort to a non-probabilistic, discriminative view:

$$s_{NN}(\mathbf{a}|\mathbf{e}, \mathbf{f}) = \prod_{i=1}^{|\mathbf{e}|} t_{lex}(e_i, f_{a_i}|\mathbf{e}, \mathbf{f})t_d(a_i, a_{i-1}|\mathbf{e}, \mathbf{f}) \quad (5)$$

where  $t_{lex}$  is a lexical translation score computed by neural network, and  $t_d$  is a distortion score.

In the classic HMM word alignment model, context is not considered in the lexical translation probability. Although we can rewrite  $P_{lex}(e_i|f_{a_i})$  to  $P_{lex}(e_i|\text{context of } f_{a_i})$  to model context, it introduces too many additional parameters and leads to serious over-fitting problem due to data sparseness. As a matter of fact, even without any contexts, the lexical translation table in HMM already contains  $O(|V_e| * |V_f|)$  parameters, where  $|V_e|$  and  $V_f$  denote source and target vocabulary sizes. In contrast, our model does not maintain a separate translation score parameters for every source-target word pair, but computes  $t_{lex}$  through a multi-layer network, which naturally handles contexts on both sides without explosive growth of number of parameters.

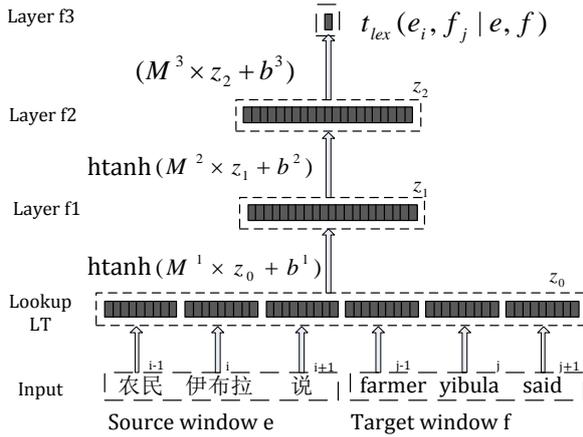


Figure 2: Network structure for computing context dependent lexical translation scores. The example computes translation score  $b^3$  for word pair  $(yibula, yibulayin)$  given its surrounding context.

Figure 2 shows the neural network we used to compute context dependent lexical translation score  $t_{lex}$ . For word pair  $(e_i, f_j)$ , we take fixed length windows surrounding both  $e_i$  and  $f_j$  as input:  $(e_{i-\frac{sw}{2}}, \dots, e_{i+\frac{sw}{2}}, f_{j-\frac{tw}{2}}, \dots, f_{j+\frac{tw}{2}})$ , where  $sw, tw$  stand window sizes on source and target side respectively. Words are converted to embeddings using the lookup table  $LT$ , and the catenation of embeddings are fed to a classic neural network with two hidden-layers, and the output of the network is the our lexical translation score:

$$t_{lex}(e_i, f_j | \mathbf{e}, \mathbf{f}) = f_3 \circ f_2 \circ f_1 \circ LT(window(e_i), window(f_j)) \quad (6)$$

$f_1$  and  $f_2$  layers use  $htanh$  as activation functions, while  $f_3$  is only a linear transformation with no activation function.

For the distortion  $t_d$ , we could use a lexicalized distortion model:

$$t_d(a_i, a_{i-1} | \mathbf{e}, \mathbf{f}) = t_d(a_i - a_{i-1} | window(f_{a_{i-1}})) \quad (7)$$

which can be computed by a neural network similar to the one used to compute lexical translation scores. If we map jump distance  $(a_i - a_{i-1})$  to  $B$  buckets, we can change the length of the output layer to  $B$ , where each dimension in the output stands for a different bucket of jump distances. But we found in our initial experiments on small scale data, lexicalized distortion does not produce better alignment over the simple jump-distance based model. So we drop the lexicalized

distortion and reverse to the simple version:

$$t_d(a_i, a_{i-1} | \mathbf{e}, \mathbf{f}) = t_d(a_i - a_{i-1}) \quad (8)$$

Vocabulary  $V$  of our alignment model consists of a source vocabulary  $V_e$  and a target vocabulary  $V_f$ . As in (Collobert et al., 2011), in addition to real words, each vocabulary contains a special unknown word symbol  $\langle unk \rangle$  to handle unseen words; two sentence boundary symbols  $\langle s \rangle$  and  $\langle /s \rangle$ , which are filled into surrounding window when necessary; furthermore, to handle null alignment, we must also include a special null symbol  $\langle null \rangle$ . When  $f_j$  is null word, we simply fill the surrounding window with the identical null symbols.

To decode our model, the lexical translation scores are computed for each source-target word pair in the sentence pair, which requires going through the neural network  $(|\mathbf{e}| \times |\mathbf{f}|)$  times; after that, the *forward-backward* algorithm can be used to find the viterbi path as in the classic HMM model.

The majority of tunable parameters in our model resides in the lookup table  $LT$ , which is a  $(L \times (|V_e| + |V_f|))$ -dimension matrix. For a reasonably large vocabulary, the number is much smaller than the number of parameters in classic HMM model, which is in the order of  $(|V_e| \times |V_f|)$ .<sup>1</sup>

The ability to model context is not unique to our model. In fact, discriminative word alignment can model contexts by deploying arbitrary features (Moore, 2005). Different from previous discriminative word alignment, our model does not use manually engineered features, but learn “features” automatically from raw words by the neural network. (Berger et al., 1996) use a maximum entropy model to model the bag-of-words context for word alignment, but their model treats each word as a distinct feature, which can not leverage the similarity between words as our model.

## 5 Training

Although unsupervised training technique such as *Contrastive Estimation* as in (Smith and Eisner, 2005), (Dyer et al., 2011) can be adapted to train

<sup>1</sup>In practice, the number of non-zero parameters in classic HMM model would be much smaller, as many words do not co-occur in bilingual sentence pairs. In our experiments, the number of non-zero parameters in classic HMM model is about 328 millions, while the NN model only has about 4 millions.

our model from raw sentence pairs, they are too computational demanding as the lexical translation probabilities must be computed from neural networks. Hence, we opt for a simpler supervised approach, which learns the model from sentence pairs with word alignment. As we do not have a large manually word aligned corpus, we use traditional word alignment models such as HMM and IBM model 4 to generate word alignment on a large parallel corpus. We obtain bi-directional alignment by running the usual *grow-diag-final* heuristics (Koehn et al., 2003) on unidirectional results from both directions, and use the results as our training data. Similar approach has been taken in speech recognition task (Dahl et al., 2012), where training data for neural network model is generated by forced decoding with traditional Gaussian mixture models.

Tunable parameters in neural network alignment model include: word embeddings in lookup table  $LT$ , parameters  $W^l, b^l$  for linear transformations in the hidden layers of the neural network, and distortion parameters  $s_d$  of jump distance. We take the following ranking loss with margin as our training criteria:

$$loss(\theta) = \sum_{\text{every } (\mathbf{e}, \mathbf{f})} \max\{0, 1 - s_\theta(\mathbf{a}^+ | \mathbf{e}, \mathbf{f}) + s_\theta(\mathbf{a}^- | \mathbf{e}, \mathbf{f})\} \quad (9)$$

where  $\theta$  denotes all tunable parameters,  $\mathbf{a}^+$  is the gold alignment path,  $\mathbf{a}^-$  is the highest scoring incorrect alignment path under  $\theta$ , and  $s_\theta$  is model score for alignment path defined in Eq. 5. One nuance here is that the gold alignment after *grow-diag-final* contains many-to-many links, which cannot be generated by any path. Our solution is that for each source word alignment multiple target, we randomly choose one link among all candidates as the golden link.

Because our multi-layer neural network is inherently non-linear and is non-convex, directly training against the above criteria is unlikely to yield good results. Instead, we take the following steps to train our model.

### 5.1 Pre-training initial word embedding with monolingual data

Most parameters reside in the word embeddings. To get a good initial value, the usual approach is to pre-train the embeddings on a large monolingual corpus. We replicate the work in (Collobert

et al., 2011) and train word embeddings for source and target languages from their monolingual corpus respectively. Our vocabularies  $V_s$  and  $V_t$  contain the most frequent 100,000 words from each side of the parallel corpus, and all other words are treated as unknown words. We set word embedding length to 20, window size to 5, and the length of the only hidden layer to 40. Follow (Turian et al., 2010), we randomly initialize all parameters to  $[-0.1, 0.1]$ , and use stochastic gradient descent to minimize the ranking loss with a fixed learning rate 0.01. Note that embedding for null word in either  $V_e$  and  $V_f$  cannot be trained from monolingual corpus, and we simply leave them at the initial value untouched.

Word embeddings from monolingual corpus learn strong syntactic knowledge of each word, which is not always desirable for word alignment between some language pairs like English and Chinese. For example, many Chinese words can act as a verb, noun and adjective without any change, while their English counter parts are distinct words with quite different word embeddings due to their different syntactic roles. Thus we have to modify the word embeddings in subsequent steps according to bilingual data.

### 5.2 Training neural network based on local criteria

Training the network against the sentence level criteria Eq. 5 directly is not efficient. Instead, we first ignore the distortion parameters and train neural networks for lexical translation scores against the following local pairwise loss:

$$\max\{0, 1 - t_\theta((e, f)^+ | \mathbf{e}, \mathbf{f}) + t_\theta((e, f)^- | \mathbf{e}, \mathbf{f})\} \quad (10)$$

where  $(e, f)^+$  is a correct word pair,  $(e, f)^-$  is a wrong word pair in the same sentence, and  $t_\theta$  is as defined in Eq. 6. This training criteria essentially means our model suffers loss unless it gives correct word pairs a higher score than random pairs from the same sentence pair with some margin.

We initialize the lookup table with embeddings obtained from monolingual training, and randomly initialize all  $W^l$  and  $b^l$  in linear layers to  $[-0.1, 0.1]$ . We minimize the loss using stochastic gradient descent as follows. We randomly cycle through all sentence pairs in training data; for each correct word pair (including null alignment), we generate a positive example, and generate two negative examples by randomly corrupting either

side of the pair with another word in the sentence pair. We set learning rate to 0.01. As there is no clear stopping criteria, we simply run the stochastic optimizer through parallel corpus for  $N$  iterations. In this work,  $N$  is set to 50.

To make our model concrete, there are still hyper-parameters to be determined: the window size  $sw$  and  $tw$ , the length of each hidden layer  $L_l$ . We empirically set  $sw$  and  $tw$  to 11,  $L_1$  to 120, and  $L_2$  to 10, which achieved a minimal loss on a small held-out data among several settings we tested.

### 5.3 Training distortion parameters

We fix neural network parameters obtained from the last step, and tune the distortion parameters  $s_d$  with respect to the sentence level loss using standard stochastic gradient descent. We use a separate parameter for jump distance from -7 and 7, and another two parameters for longer forward/backward jumps. We initialize all parameters in  $s_d$  to 0, set the learning rate for the stochastic optimizer to 0.001. As there are only 17 parameters in  $s_d$ , we only need to run the optimizer over a small portion of the parallel corpus.

### 5.4 Tuning neural network based on sentence level criteria

Up-to-now, parameters in the lexical translation neural network have not been trained against the sentence level criteria Eq. 5. We could achieve this by re-using the same online training method used to train distortion parameters, except that we now fix the distortion parameters and let the loss back-propagate through the neural networks. Sentence level training does not take larger context in modeling word translations, but only to optimize the parameters regarding to the sentence level loss. This tuning is quite slow, and it did not improve alignment on an initial small scale experiment; so, we skip this step in all subsequent experiment in this work.

## 6 Experiments and Results

We conduct our experiment on Chinese-to-English word alignment task. We use the manually aligned Chinese-English alignment corpus (Haghighi et al., 2009) which contains 491 sentence pairs as test set. We adapt the segmentation on the Chinese side to fit our word segmentation standard.

### 6.1 Data

Our parallel corpus contains about 26 million unique sentence pairs in total which are mined from web.

The monolingual corpus to pre-train word embeddings are also crawled from web, which amounts to about 1.1 billion unique sentences for English and about 300 million unique sentences for Chinese. As pre-processing, we lowercase all English words, and map all numbers to one special token; and we also map all email addresses and URLs to another special token.

### 6.2 Settings

We use classic HMM and IBM model 4 as our baseline, which are generated by Giza++ (Och and Ney, 2000). We train our proposed model from results of classic HMM and IBM model 4 separately. Since classic HMM, IBM model 4 and our model are all uni-directional, we use the standard *grow-diag-final* to generate bi-directional results for all models.

Models are evaluated on the manually aligned test set using standard metric: precision, recall and F1-score.

### 6.3 Alignment Result

It can be seen from Table 1, the proposed model consistently outperforms its corresponding baseline whether it is trained from alignment of classic HMM or IBM model 4. It is also clear that the

setting	prec.	recall	F-1
HMM	0.768	0.786	0.777
HMM+NN	0.810	0.790	0.798
IBM4	0.839	0.805	0.822
IBM4+NN	<b>0.885</b>	<b>0.812</b>	<b>0.847</b>

Table 1: Word alignment result. The first row and third row show baseline results obtained by classic HMM and IBM4 model. The second row and fourth row show results of the proposed model trained from HMM and IBM4 respectively.

results of our model also depends on the quality of baseline results, which is used as training data of our model. In future we would like to explore whether our method can improve other word alignment models.

We also conduct experiment to see the effect on end-to-end SMT performance. We train hier-

archical phrase model (Chiang, 2007) from different word alignments. Despite different alignment scores, we do not obtain significant difference in translation performance. In our C-E experiment, we tuned on NIST-03, and tested on NIST-08. Case-insensitive BLEU-4 scores on NIST-08 test are 0.305 and 0.307 for models trained from IBM-4 and NN alignment results. The result is not surprising considering our parallel corpus is quite large, and similar observations have been made in previous work as (DeNero and Macherey, 2011) that better alignment quality does not necessarily lead to better end-to-end result.

## 6.4 Result Analysis

### 6.4.1 Error Analysis

From Table 1 we can see higher F-1 score of our model mainly comes from higher precision, with recall similar to baseline. By analyzing the results, we found out that for both baseline and our model, a large part of missing alignment links involves stop words like English words “the”, “a”, “it” and Chinese words “de”. Stop words are inherently hard to align, which often requires grammatical judgment unavailable to our models; as they are also extremely frequent, our model fully learns their alignment patterns of the baseline models, including errors. On the other hand, our model performs better on low-frequency words, especially proper nouns. Take person names for example. Most names are low-frequency words, on which baseline HMM and IBM4 models show the “garbage collector” phenomenon. In our model, different person names have very similar word embeddings on both English side and Chinese side, due to monolingual pre-training; what is more, different person names often appear in similar contexts. As our model considers both word embeddings and contexts, it learns that English person names should be aligned to Chinese person names, which corrects errors of baseline models and leads to better precision.

### 6.4.2 Effect of context

To examine how context contribute to alignment quality, we re-train our model with different window size, all from result of IBM model 4. From Figure 3, we can see introducing context increase the quality of the learned alignment, but the benefit is diminished for window size over 5. On the other hand, the results are quite stable even with large window size 13, without noticeable over-

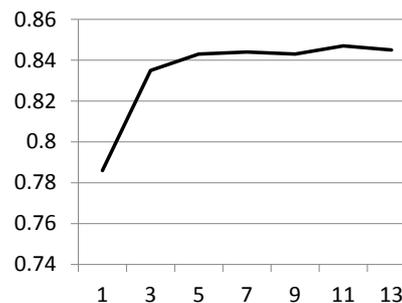


Figure 3: Effect of different window sizes on word alignment F-score.

fitting problem. This is not surprising considering that larger window size only requires slightly more parameters in the linear layers. Lastly, it is worth noticing that our model with no context (window size 1) performs much worse than settings with larger window size and baseline IBM4. Our explanation is as follows. Our model uses the simple jump distance based distortion, which is weaker than the more sophisticated distortions in IBM model 4; thus without context, it does not perform well compared to IBM model 4. With larger window size, our model is able to produce more accurate translation scores based on more contexts, which leads to better alignment despite the simpler distortions.

IBM4+NN	F-1
1-hidden-layer	0.834
2-hidden-layer	0.847
3-hidden-layer	0.843

Table 3: Effect of different number of hidden layers. Two hidden layers outperform one hidden layer, while three hidden layers do not bring further improvement.

### 6.4.3 Effect of number of hidden layers

Our neural network contains two hidden layers besides the lookup layer. It is natural to ask whether adding more layers would be beneficial. To answer this question, we train models with 1, 2 and 3 layers respectively, all from result of IBM model 4. For 1-hidden-layer setting, we set the hidden layer length to 120; and for 3-hidden-layer setting, we set hidden layer lengths to 120, 100, 10 respectively. As can be seen from Table 3, 2-hidden-layer outperforms the 1-hidden-layer setting, while another hidden layer does not bring

word	good	history	british	served	labs	zetian	laggards
LM	bad	tradition	russian	worked	networks	hongzhang	underperformers
	great	culture	japanese	lived	technologies	yaobang	transferees
	strong	practice	dutch	offered	innovations	keming	megabanks
	true	style	german	delivered	systems	xingzhi	mutuals
	easy	literature	canadian	produced	industries	ruihua	non-starters
WA	nice	historical	uk	offering	lab	hongzhang	underperformers
	great	historic	britain	serving	laboratories	qichao	illiterates
	best	developed	english	serve	laboratory	xueqin	transferees
	pretty	record	classic	delivering	exam	fuhuan	matriculants
	excellent	recording	england	worked	experiments	bingkun	megabanks

Table 2: Nearest neighbors of several words according to their embedding distance. *LM* shows neighbors of word embeddings trained by monolingual language model method; *WA* shows neighbors of word embeddings trained by our word alignment model.

improvement. Due to time constraint, we have not tuned the hyper-parameters such as length of hidden layers in 1 and 3-hidden-layer settings, nor have we tested settings with more hidden-layers. It would be wise to test more settings to verify whether more layers would help.

#### 6.4.4 Word Embedding

Following (Collobert et al., 2011), we show some words together with its nearest neighbors using the Euclidean distance between their embeddings. As we can see from Table 2, after bilingual training, “bad” is no longer in the nearest neighborhood of “good” as they hold opposite semantic meanings; the nearest neighbor of “history” is now changed to its related adjective “historical”. Neighbors of proper nouns such as person names are relatively unchanged. For example, neighbors of word “zetian” are all Chinese names in both settings. As Chinese language lacks morphology, the single form and plural form of a noun in English often correspond to the same Chinese word, thus it is desirable that the two English words should have similar word embeddings. While this is true for relatively frequent nouns such as “lab” and “labs”, rarer nouns still remain near their monolingual embeddings as they are only modified a few times during the bilingual training. As shown in last column, neighborhood of “laggards” still consists of other plural forms even after bilingual training.

## 7 Conclusion

In this paper, we explore applying deep neural network for word alignment task. Our model

integrates a multi-layer neural network into an HMM-like framework, where context dependent lexical translation score is computed by neural network, and distortion is modeled by a simple jump-distance scheme. Our model is discriminatively trained on bilingual corpus, while huge monolingual data is used to pre-train word-embeddings. Experiments on large-scale Chinese-to-English task show that the proposed method produces better word alignment results, compared with both classic HMM model and IBM model 4.

For future work, we will investigate more settings of different hyper-parameters in our model. Secondly, we want to explore the possibility of unsupervised training of our neural word alignment model, without reliance of alignment result of other models. Furthermore, our current model use rather simple distortions; it might be helpful to use more sophisticated model such as ITG (Wu, 1997), which can be modeled by Recursive Neural Networks (Socher et al., 2011).

## Acknowledgments

We thank anonymous reviewers for insightful comments. We also thank Dongdong Zhang, Lei Cui, Chunyang Wu and Zhenyan He for fruitful discussions.

## References

- Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Frédéric Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. *Innovations in Machine Learning*, pages 137–186.
- Yoshua Bengio, Pascal Lamblin, Dan Popovici, and

- Hugo Larochelle. 2007. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153.
- Yoshua Bengio. 2009. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*, 2(1):1–127.
- Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March.
- JS Bridle. 1990. Neurocomputing: Algorithms, architectures and applications, chapter probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition.
- Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.
- David Chiang. 2007. Hierarchical phrase-based translation. *computational linguistics*, 33(2):201–228.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.
- George E Dahl, Dong Yu, Li Deng, and Alex Acero. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(1):30–42.
- John DeNero and Klaus Macherey. 2011. Model-based aligner combination using dual decomposition. In *Proc. ACL*.
- Chris Dyer, Jonathan Clark, Alon Lavie, and Noah A Smith. 2011. Unsupervised word alignment with arbitrary features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 409–419. Association for Computational Linguistics.
- Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised itg models. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 923–931. Association for Computational Linguistics.
- Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.
- Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu, and Yann LeCun. 2010. Learning convolutional feature hierarchies for visual recognition. *Advances in Neural Information Processing Systems*, pages 1090–1098.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.
- Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- Yann LeCun. 1985. A learning scheme for asymmetric threshold networks. *Proceedings of Cognitive*, 85:599–604.
- Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. 2007. Efficient sparse coding algorithms. *Advances in neural information processing systems*, 19:801.
- Shujie Liu, Chi-Ho Li, and Ming Zhou. 2010. Discriminative pruning for discriminative itg alignment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL*, volume 10, pages 316–324.
- Y MarcAurelio Ranzato, Lan Boureau, and Yann LeCun. 2007. Sparse feature learning for deep belief networks. *Advances in neural information processing systems*, 20:1185–1192.
- Robert C Moore. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 81–88. Association for Computational Linguistics.
- Jan Niehues and Alex Waibel. 2012. Continuous space language models using restricted boltzmann machines. In *Proceedings of the ninth International Workshop on Spoken Language Translation (IWSLT)*.
- Franz Josef Och and Hermann Ney. 2000. Giza++: Training of statistical translation models.
- Frank Seide, Gang Li, and Dong Yu. 2011. Conversational speech transcription using context-dependent deep neural networks. In *Proc. Interspeech*, pages 437–440.

- Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.
- Richard Socher, Cliff C Lin, Andrew Y Ng, and Christopher D Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, volume 2, page 7.
- Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics.
- Le Hai Son, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 39–48. Association for Computational Linguistics.
- Ivan Titov, Alexandre Klementiev, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. *Urbana*, 51:61801.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 836–841. Association for Computational Linguistics.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics*, 23(3):377–403.

# Microblogs as Parallel Corpora

Wang Ling<sup>123</sup> Guang Xiang<sup>2</sup> Chris Dyer<sup>2</sup> Alan Black<sup>2</sup> Isabel Trancoso<sup>13</sup>

(1)L<sup>2</sup>F Spoken Systems Lab, INESC-ID, Lisbon, Portugal

(2)Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, USA

(3)Instituto Superior Técnico, Lisbon, Portugal

{lingwang, guangx, cdyer, awb}@cs.cmu.edu

isabel.trancoso@inesc-id.pt

## Abstract

In the ever-expanding sea of microblog data, there is a surprising amount of naturally occurring parallel text: some users create post multilingual messages targeting international audiences while others “retweet” translations. We present an efficient method for detecting these messages and extracting parallel segments from them. We have been able to extract over 1M Chinese-English parallel segments from Sina Weibo (the Chinese counterpart of Twitter) using only their public APIs. As a supplement to existing parallel training data, our automatically extracted parallel data yields substantial translation quality improvements in translating microblog text and modest improvements in translating edited news commentary. The resources in described in this paper are available at <http://www.cs.cmu.edu/~lingwang/utopia>.

## 1 Introduction

Microblogs such as Twitter and Facebook have gained tremendous popularity in the past 10 years. In addition to being an important form of communication for many people, they often contain extremely current, even breaking, information about world events. However, the writing style of microblogs tends to be quite colloquial, with frequent orthographic innovation (*R U still with me or what?*) and nonstandard abbreviations (*idk! shm*)—quite unlike the style found in more traditional, edited genres. This poses considerable problems for traditional NLP tools, which were developed with other domains in mind, which often make strong assumptions about orthographic uniformity (i.e., there is just one way to spell *you*). One approach to cope with this problem is to annotate in-domain data (Gimpel et al., 2011).

Machine translation suffers acutely from the domain-mismatch problem caused by microblog text. On one hand, standard models are probably suboptimal since they (like many models) assume orthographic uniformity in the input. However, more acutely, the data used to develop these systems and train their models is drawn from formal and carefully edited domains, such as parallel web pages and translated legal documents. MT training data seldom looks anything like microblog text.

This paper introduces a method for finding naturally occurring parallel microblog text, which helps address the domain-mismatch problem. Our method is inspired by the perhaps surprising observation that a reasonable number of microblog users tweet “in parallel” in two or more languages. For instance, the American entertainer Snoop Dogg regularly posts parallel messages on Sina Weibo (Mainland China’s equivalent of Twitter), for example, *watup Kenny Mayne!! - Kenny Mayne, 最近这么样啊!!*, where an English message and its Chinese translation are in the same post, separated by a dash. Our method is able to identify and extract such translations. Briefly, this requires determining if a tweet contains more than one language, if these multilingual utterances contain translated material (or are due to something else, such as code switching), and what the translated spans are.

The paper is organized as follows. Section 2 describes the related work in parallel data extraction. Section 3 presents our model to extract parallel data within the same document. Section 4 describes our extraction pipeline. Section 5 describes the data we gathered from both Sina Weibo (Chinese-English) and Twitter (Chinese-English and Arabic-English). We then present experiments showing that our harvested data not only substantially improves translations of microblog text with

existing (and arguably inappropriate) translation models, but that it improves the translation of more traditional MT genres, like newswire. We conclude in Section 6.

## 2 Related Work

Automatic collection of parallel data is a well-studied problem. Approaches to finding parallel web documents automatically have been particularly important (Resnik and Smith, 2003; Fukushima et al., 2006; Li and Liu, 2008; Uszkoreit et al., 2010; Ture and Lin, 2012). These broadly work by identifying promising candidates using simple features, such as URL similarity or “gist translations” and then identifying truly parallel segments with more expensive classifiers. More specialized resources were developed using manual procedures to leverage special features of very large collections, such as Europarl (Koehn, 2005).

Mining parallel or comparable messages from microblogs has mainly relied on Cross-Lingual Information Retrieval techniques (CLIR). Jelh et al. (2012) attempt to find pairs of tweets in Twitter using Arabic tweets as search queries in a CLIR system. Afterwards, the model described in (Xu et al., 2001) is applied to retrieve a set of ranked translation candidates for each Arabic tweet, which are then used as parallel candidates.

The work on mining parenthetical translations (Lin et al., 2008), which attempts to find translations within the same document, has some similarities with our work, since parenthetical translations are within the same document. However, parenthetical translations are generally used to translate names or terms, which is more limited than our work which extracts whole sentence translations.

Finally, crowd-sourcing techniques to obtain translations have been previously studied and applied to build datasets for casual domains (Zbib et al., 2012; Post et al., 2012). These approaches require remunerated workers to translate the messages, and the amount of messages translated per day is limited. We aim to propose a method that acquires large amounts of parallel data for free. The drawback is that there is a margin of error in the parallel segment identification and alignment. However, our system can be tuned for precision or for recall.

## 3 Parallel Segment Retrieval

We will first abstract from the domain of Microblogs and focus on the task of retrieving parallel segments from single documents. Prior work on finding parallel data attempts to reason about the probability that pairs of documents  $(\mathbf{x}, \mathbf{y})$  are parallel. In contrast, we only consider one document at a time, defined by  $\mathbf{x} = x_1, x_2, \dots, x_n$ , and consisting of  $n$  tokens, and need to determine whether there is **parallel data** in  $\mathbf{x}$ , and if so, where are the parallel **segments** and their **languages**. For simplicity, we assume that there are at most 2 continuous segments that are parallel.

As representation for the parallel segments within the document, we use the tuple  $([p, q], l, [u, v], r, \mathbf{a})$ . The word indexes  $[p, q]$  and  $[u, v]$  are used to identify the left segment (from  $p$  to  $q$ ) and right segment (from  $u$  to  $v$ ), which are parallel. We shall refer  $[p, q]$  and  $[u, v]$  as the **spans** of the left and right segments. To avoid overlaps, we set the constraint  $p \leq q < u \leq v$ . Then, we use  $l$  and  $r$  to identify the language of the left and right segments, respectively. Finally,  $\mathbf{a}$  represents the word alignment between the words in the left and the right segments.

The main problem we address is to find the parallel data when the boundaries of the parallel segments are not defined explicitly. If we knew the indexes  $[p, q]$  and  $[u, v]$ , we could simply run a language detector for these segments to find  $l$  and  $r$ . Then, we would use an word alignment model (Brown et al., 1993; Vogel et al., 1996), with source  $\mathbf{s} = x_p, \dots, x_q$ , target  $\mathbf{t} = x_u, \dots, x_v$  and lexical table  $\theta_{l,r}$  to calculate the Viterbi alignment  $\mathbf{a}$ . Finally, from the probability of the word alignments, we can determine whether the segments are parallel.

Thus, our model will attempt to find the optimal values for the segments  $[p, q][u, v]$ , languages  $l, r$  and word alignments  $\mathbf{a}$  jointly. However, there are two problems with this approach. Firstly, word alignment models generally attribute higher probabilities to smaller segments, since these are the result of a smaller product chain of probabilities. In fact, because our model can freely choose the segments to align, choosing only one word as the left segment that is well aligned to a word in the right segment would be the best choice. This is obviously not our goal, since we would not obtain any useful sentence pairs. Secondly, inference must be performed over the combination of all latent variables, which is intractable using

a brute force algorithm. We shall describe our model to solve the first problem in 3.1 and our dynamic programming approach to make the inference tractable in 3.2.

### 3.1 Model

We propose a simple (non-probabilistic) three-factor model that models the spans of the parallel segments, their languages, and word alignments jointly. This model is defined as follows:

$$S([u, v], r, [p, q], l, \mathbf{a} \mid \mathbf{x}) = S_S^\alpha([p, q], [u, v] \mid \mathbf{x}) \times S_L^\beta(l, r \mid [p, q], [u, v], \mathbf{x}) \times S_T^\gamma(\mathbf{a} \mid [p, q], l, [u, v], r, \mathbf{x})$$

Each of the components is weighted by the parameters  $\alpha$ ,  $\beta$  and  $\gamma$ . We set these values empirically  $\alpha = 0.3$ ,  $\beta = 0.3$  and  $\gamma = 0.4$ , and leave the optimization of these parameters as future work. We discuss the components of this model in turn.

**Span score  $S_S$ .** We define the score of hypothesized pair of spans  $[p, q]$ ,  $[u, v]$  as:

$$S_S([p, q], [u, v] \mid \mathbf{x}) = \frac{(q - p + 1) + (v - u + 1)}{\sum_{0 < p' \leq q' < u' \leq v' \leq n} (q' - p' + 1) + (v' - u' + 1)} \times \psi([p, q], [u, v], \mathbf{x})$$

The first factor is a distribution over all spans that assigns higher probability to segmentations that cover more words in the document. It is highest for segmentations that cover all the words in the document (this is desirable since there are many sentence pairs that can be extracted but we want to find the largest sentence pair in the document). The function  $\psi$  takes on values of 0 or 1 depending on whether certain constraints are violated, these include: parenthetical constraints that enforce that spans must not break text within parenthetical characters and language constraints that ensure that we do not break a sequence of Mandarin characters, Arabic words or Latin words.

**Language score  $S_L$ .** The language score  $S_L(l, r \mid [p, q], [u, v], \mathbf{x})$  indicates whether the language labels  $l, r$  are appropriate to the document contents:

$$S_L(l, r \mid [p, q], [u, v], \mathbf{x}) = \frac{\sum_{i=p}^q L(l, x_i) + \sum_{i=u}^v L(r, x_i)}{n}$$

where  $L(l, x)$  is a language detection function that yields 1 if the word  $x_i$  is in language  $l$ , and 0 otherwise. We build the function simply by considering all words that are composed of Latin characters as English, Arabic characters as Arabic and Han characters as Mandarin. This approach is not perfect, but it is simple and works reasonably well for our purposes.

**Translation score  $S_T$ .** The translation score  $S_T(\mathbf{a} \mid [p, q], l, [u, v], r)$  indicates whether  $[p, q]$  is a reasonable translation of  $[u, v]$  with the alignment  $\mathbf{a}$ . We rely on IBM Model 1 probabilities for this score:

$$S_T(\mathbf{a} \mid [p, q], l, [u, v], r, \mathbf{x}) = \frac{1}{(q - p + 1)^{v-u+2}} \prod_{i=u}^v P_{M1}(x_i \mid x_{a_i}).$$

The lexical tables  $P_{M1}$  for the various language pairs are trained a priori using available parallel corpora. While IBM Model 1 produces worse alignments than other models, in our problem, we need to efficiently consider all possible spans, language pairs and word alignments, which makes the problem intractable. We will show that dynamic programming can be used to make this problem tractable, using Model 1. Furthermore, IBM Model 1 has shown good performance for sentence alignment systems previously (Xu et al., 2005; Braune and Fraser, 2010).

### 3.2 Inference

Our goal is to find the spans, language pair and alignments such that:

$$\arg \max_{[p, q], l, [u, v], r, \mathbf{a}} S([p, q], l, [u, v], r, \mathbf{a} \mid \mathbf{x}) \quad (1)$$

A high score indicates that the predicted bispan is likely to correspond to a valid parallel span, so we set a constant threshold  $\tau$  to determine whether a document has parallel data, i.e., the value of  $z$ :

$$z^* = \max_{[u, v], r, [p, q], l, \mathbf{a}} S([u, v], r, [p, q], l, \mathbf{a} \mid \mathbf{x}) > \tau$$

Naively maximizing Eq. 1 would require  $O(|\mathbf{x}|^6)$  operations, which is too inefficient to be practical on large datasets. To process millions of documents, this process would need to be optimized.

The main bottleneck of the naive algorithm is finding new Viterbi Model 1 word alignments every time we change the spans. Thus, we propose

an iterative approach to compute the Viterbi word alignments for IBM Model 1 using dynamic programming.

**Dynamic programming search.** The insight we use to improve the runtime is that the Viterbi word alignment of a bispan can be reused to calculate the Viterbi word alignments of larger bispans. The algorithm operates on a 4-dimensional chart of bispans. It starts with the minimal valid span (i.e.,  $[0, 0], [1, 1]$ ) and progressively builds larger spans from smaller ones. Let  $A_{p,q,u,v}$  represent the Viterbi alignment (under  $S_T$ ) of the bispan  $[p, q], [u, v]$ . The algorithm uses the following recursions defined in terms of four operations  $\lambda_{\{+v,+u,+p,+q\}}$  that manipulate a single dimension of the bispan to construct larger spans:

- $A_{p,q,u,v+1} = \lambda_{+v}(A_{p,q,u,v})$  adds one token to the end of the right span with index  $v + 1$  and find the viterbi alignment for that token. This requires iterating over all the tokens in the left span,  $[p, q]$  and possibly updating their alignments. See Fig. 1 for an illustration.
- $A_{p,q,u+1,v} = \lambda_{+u}(A_{p,q,u,v})$  removes the first token of the right span with index  $u$ , so we only need to remove the alignment from  $u$ , which can be done in time  $O(1)$ .
- $A_{p,q+1,u,v} = \lambda_{+q}(A_{p,q,u,v})$  adds one token to the end of the left span with index  $q + 1$ , we need to check for each word in the right span, if aligning to the word in index  $q+1$  yields a better translation probability. This update requires  $n - q + 1$  operations.
- $A_{p+1,q,u,v} = \lambda_{+p}(A_{p,q,u,v})$  removes the first token of the left span with index  $p$ . After removing the token, we need to find new alignments for all tokens that were aligned to  $p$ . Thus, the number of operations for this update is  $K \times (q - p + 1)$ , where  $K$  is the number of words that were aligned to  $p$ . In the best case, no words are aligned to the token in  $p$ , and we can simply remove it. In the worst case, if all target words were aligned to  $p$ , this update will result in the recalculation of all Viterbi Alignments.

The algorithm proceeds until all valid cells have been computed. One important aspect is that the update functions differ in complexity, so the sequence of updates we apply will impact the performance of the system. Most spans are reachable using any of the four update functions. For instance, the span  $A_{2,3,4,5}$  can be reached using  $\lambda_{+v}(A_{2,3,4,4})$ ,  $\lambda_{+u}(A_{2,3,3,5})$ ,  $\lambda_{+q}(A_{2,2,4,5})$  or  $\lambda_{+p}(A_{1,3,4,5})$ . However, we want to use  $\lambda_{+u}$

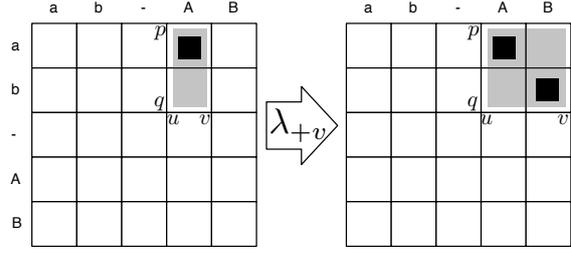


Figure 1: Illustration of the  $\lambda_{+v}$  operator. The light gray boxes show the parallel span and the dark boxes show the span’s Viterbi alignment. In this example, the parallel message contains a “translation” of a b to A B.

whenever possible, since it only requires one operation, although that is not always possible. For instance, the state  $A_{2,2,2,4}$  cannot be reached using  $\lambda_{+u}$ , since the state  $A_{2,2,1,4}$  is not valid, because the spans overlap. If this happens, incrementally more expensive updates need to be used, such as  $\lambda_{+v}$ , then  $\lambda_{+q}$ , which are in the same order of complexity. Finally, we want to minimize the use of  $\lambda_{+p}$ , which is quadratic in the worst case. Thus, we use the following recursive formulation that guarantees the optimal outcome:

$$A_{p,q,u,v} = \begin{cases} \lambda_{+u}(A_{p,q,u-1,v}) & \text{if } u > q + 1 \\ \lambda_{+v}(A_{p,q,u,v-1}) & \text{else if } v > q + 1 \\ \lambda_{+p}(A_{p-1,q,u,v}) & \text{else if } q = p + 1 \\ \lambda_{+q}(A_{p,q-1,u,v}) & \text{otherwise} \end{cases}$$

This transition function applies the cheapest possible update to reach state  $A_{p,q,u,v}$ .

**Complexity analysis.** We can see that  $\lambda_{+u}$  is only needed in the following the cases  $[0, 1][2, 2], [1, 2][3, 3], \dots, [n - 2, n - 1][n, n]$ . Since, this update is quadratic in the worst case, the complexity of this operations is  $O(n^3)$ . The update  $\lambda_{+q}$ , is applied to the cases  $[*, 1][2, 2], [*, 2][3, 3], \dots, [*, n - 1], [n, n]$ , where  $*$  denotes any number within the span constraints but not present in previous updates. Since, the update is linear and we need to iterate through all tokens twice, this update takes  $O(n^3)$  operations. The update  $\lambda_{+v}$  is applied for the cases  $[*, 1][2, *], [*, 2][3, *], \dots, [*, n - 1], [n, *]$ . Thus, with three degrees of freedom and a linear update, it runs in  $O(n^4)$  time. Finally, update  $\lambda_{+u}$  runs in constant time, but is run for all remaining cases, which constitute  $O(n^4)$  space. By summing the

executions of all updates, we observe that the order of magnitude of our exact inference process is  $O(n^4)$ . Note that for exact inference, it is not possible to get a lower order of magnitude, since we need to at least iterate through all possible span values once, which takes  $O(n^4)$  time.

## 4 Parallel Data Extraction

We will now describe our method to extract parallel data from Microblogs. The target domains in this work are Twitter and Sina Weibo, and the main language pair is Chinese-English. Furthermore, we also run the system for the Arabic-English language pair using the Twitter data.

For the Twitter domain, we use a previously crawled dataset from the years 2008 to 2013, where one million tweets are crawled every day. In total, we processed 1.6 billion tweets.

Regarding Sina Weibo, we built a crawler that continuously collects tweets from Weibo. We start from one seed user and collect his posts, and then we find the users he follows that we have not considered, and repeat. Due to the rate limiting established by the Weibo API<sup>1</sup>, we are restricted in terms of number of requests every hour, which greatly limits the amount of messages we can collect. Furthermore, each request can only fetch up to 100 posts from a user, and subsequent pages of 100 posts require additional API calls. Thus, to optimize the number of parallel posts we can collect per request, we only crawl all messages from users that have at least 10 parallel tweets in their first 100 posts. The number of parallel messages is estimated by running our alignment model, and checking if  $\tau > \phi$ , where  $\phi$  was set empirically initially, and optimized after obtaining annotated data, which will be detailed in 5.1. Using this process, we crawled 65 million tweets from Sina Weibo within 4 months.

In both cases, we first filter the collection of tweets for messages containing at least one trigram in each language of the target language pair, determined by their Unicode ranges. This means that for the Chinese-English language pair, we only keep tweets with more than 3 Mandarin characters and 3 latin words. Furthermore, based on the work in (Jelh et al., 2012), if a tweet  $A$  is identified as a retweet, meaning that it references another tweet  $B$ , we also consider the hypothesis that these tweets may be mutual translations. Thus, if  $A$  and  $B$  contain trigrams in different languages,

<sup>1</sup><http://open.weibo.com/wiki/API文档/en>

these are also considered for the extraction of parallel data. This is done by concatenating tweets  $A$  and  $B$ , and adding the constraint that  $[p, q]$  must be within  $A$  and  $[u, v]$  must be within  $B$ . Finally, identical duplicate tweets are removed.

After filtering, we obtained 1124k ZH-EN tweets from Sina Weibo, 868k ZH-EN and 136k AR-EN tweets from Twitter. These language pairs are not definite, since we simply check if there is a trigram in each language.

Finally, we run our alignment model described in section 3, and obtain the parallel segments and their scores, which measure how likely those segments are parallel. In this process, lexical tables for EN-ZH language pair used by Model 1 were built using the FBIS dataset (LDC2003E14) for both directions, a corpus of 300K sentence pairs from the news domain. Likewise, for the EN-AR language pair, we use a fraction of the NIST dataset, by removing the data originated from UN, which leads to approximately 1M sentence pairs.

## 5 Experiments

We evaluate our method in two ways. First, intrinsically, by observing how well our method identifies tweets containing parallel data, the language pair and what their spans are. Second, extrinsically, by looking at how well the data improves a translation task. This methodology is similar to that of Smith et al. (2010).

### 5.1 Parallel Data Extraction

**Data.** Our method needs to determine if a given tweet contains parallel data, and if so, what is the language pair of the data, and what segments are parallel. Thus, we had a native Mandarin speaker, also fluent in English, to annotate 2000 tweets sampled from crawled Weibo tweets. One important question of answer is what portion of the Microblogs contains parallel data. Thus, we also use the random sample Twitter and annotated 1200 samples, identifying whether each sample contains parallel data, for the EN-ZH and AR-EN filtered tweets.

**Metrics.** To test the accuracy of the score  $S$ , we ordered all 2000 samples by score. Then, we calculate the precision, recall and accuracy at increasing intervals of 10% of the top samples. We count as a true positive ( $tp$ ) if we correctly identify a parallel tweet, and as a false positive ( $fp$ ) spuriously detect a parallel tweet. Finally, a true negative ( $tn$ ) occurs when we correctly detect a non-parallel

tweet, and a false negative ( $fn$ ) if we miss a parallel tweet. Then, we set the precision as  $\frac{tp}{tp+fp}$ , recall as  $\frac{tp}{tp+fn}$  and accuracy as  $\frac{tp+tn}{tp+fp+tn+fn}$ . For language identification, we calculate the accuracy based on the number of instances that were identified with the correct language pair. Finally, to evaluate the segment alignment, we use the Word Error Rate (WER) metric, without substitutions, where we compare the left and right spans of our system and the respective spans of the reference. We count an insertion error ( $I$ ) for each word in our system’s spans that is not present in the reference span and a deletion error ( $D$ ) for each word in the reference span that is not present in our system’s spans. Thus, we set  $WER = \frac{D+I}{N}$ , where  $N$  is the number of tokens in the tweet. To compute this score for the whole test set, we compute the average of the  $WER$  for each sample.

**Results.** The precision, recall and accuracy curves are shown in Figure 2. The quality of the parallel sentence detection did not vary significantly with different setups, so we will only show the results for the best setup, which is the baseline model with span constraints.

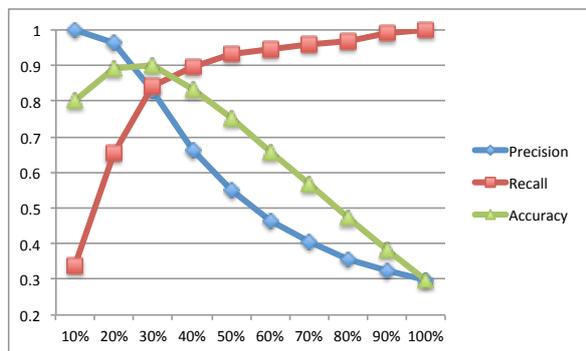


Figure 2: Precision, recall and accuracy curves for parallel data detection. The y-axis denotes the scores for each metric, and the x-axis denotes the percentage of the highest scoring sentence pairs that are kept.

From the precision and recall curves, we observe that most of the parallel data can be found at the top 30% of the filtered tweets, where 5 in 6 tweets are detected correctly as parallel, and only 1 in every 6 parallel sentences is lost. We will denote the score threshold at this point as  $\phi$ , which is a good threshold to estimate on whether the tweet is parallel. However, this parameter can be tuned for precision or recall. We also see that in total,

30% of the filtered tweets are parallel. If we generalize this ratio for the complete set with 1124k tweets, we can expect approximately 337k parallel sentences. Finally, since 65 million tweets were extracted to generate the 337k tweets, we estimate that approximately 1 parallel tweet can be found for every 200 tweets we process using our targeted approach. On the other hand, from the 1200 tweets from Twitter, we found that 27 had parallel data in the ZH-EN pair, if we extrapolate for the whole 868k filtered tweets, we expect that we can find 19530. 19530 parallel sentences from 1.6 billion tweets crawled randomly, represents 0.001% of the total corpora. For AR-EN, a similar result was obtained where we expect 12407 tweets out of the 1.6 billion to be parallel. This shows that targeted approaches can substantially reduce the crawling effort required to find parallel tweets. Still, considering that billions of tweets are posted daily, this is a substantial source of parallel data. The remainder of the tests will be performed on the Weibo dataset, which contains more parallel data. Tests on the Twitter data will be conducted as future work, when we process Twitter data on a larger scale to obtain more parallel sentences.

For the language identification task, we had an accuracy of 99.9%, since distinguishing English and Mandarin is trivial. The small percentage of errors originated from other latin languages (Ex: French) due to our naive language detector.

As for the segment alignment task. Our baseline system with no constraints obtains a WER of 12.86%, and this can be improved to 11.66% by adding constraints to possible spans. This shows that, on average, approximately 1 in 9 words on the parallel segments is incorrect. However, translation models are generally robust to such kinds of errors and can learn good translations even in the presence of imperfect sentence pairs.

Among the 578 tweets that are parallel, 496 were extracted within the same tweet and 82 were extracted from retweets. Thus, we see that the majority of the parallel data comes from within the same tweet.

**Topic analysis.** To give an intuition about the contents of the parallel data we found, we looked at the distribution over topics of the parallel dataset inferred by LDA (Blei et al., 2003). Thus, we grouped the Weibo filtered tweets by users, and ran LDA over the predicted English segments, with 12 topics. The 7 most interpretable topics are shown in Table 1. We see that the data contains a

#	Topic	Most probable words in topic
1	(Dating)	love time girl live mv back word night rt wanna
2	(Entertainment)	news video follow pong image text great day today fans
3	(Music)	cr day tour cn url amazon music full concert alive
4	(Religion)	man god good love life heart would give make lord
5	(Nightlife)	cn url beijing shanqi party adj club dj bejiner vt
6	(Chinese News)	china chinese year people world beijing years passion country government
7	(Fashion)	street fashion fall style photo men model vogue spring magazine

Table 1: Most probable words inferred using LDA in several topics from the parallel data extracted from Weibo. Topic labels (in parentheses) were assigned manually for illustration purposes.

variety of topics, both formal (Chinese news, religion) and informal (entertainment, music).

**Example sentence pairs.** To gain some perspective on the type of sentence pairs we are extracting, we will illustrate some sentence pairs we crawled and aligned automatically. Table 2 contains 5 English-Mandarin and 4 English-Arabic sentence pairs that were extracted automatically. These were chosen, since they contain some aspects that are characteristic of the text present in Microblogs and Social Media. These are:

- **Abbreviations** - In most sentence pairs examples, we can witness the use of abbreviated forms of English words, such as *wanna*, *TMI*, *4* and *imma*. These can be normalized as *want to*, *too much information*, *for* and *I am going to*, respectively. In sentence 5, we observe that this phenomena also occurs in Mandarin. We find that *TMD* is a popular way to write 他妈的 whose Pinyin rendering is *tā mā de*. The meaning of this expression depends on the context it is used, and can convey a similar connotation as adding the intensifier *the hell* to an English sentence.
- **Jargon** - Another common phenomena is the appearance of words that are only used in sub-communities. For instance, in sentence pair 4, we the jargon word *cday* is used, which is a colloquial variant for *birthday*.
- **Emoticons** - In sentence 8, we observe the presence of the emoticon *:)*, which is frequently used in this media. We found that emoticons are either translated as they are or simply removed, in most cases.
- **Syntax errors** - In the domain of microblogs, it is also common that users do not write strictly syntactic sentences, for instance, in sentence pair 7, the sentence *onni this gift only 4 u*, is clearly not syntactically correct. Firstly, *onni* is a named entity, yet it is not capitalized. Secondly, a comma should follow *onni*. Thirdly, the

verb *is* should be used after *gift*. Having examples of these sentences in the training set, with common mistakes (intentional or not), might become a key factor in training MT systems that can be robust to such errors.

- **Dialects** - We can observe a much broader range of dialects in our data, since there are no dialect standards in microblogs. For instance, in sentence pair 6, we observe an arabic word (in bold) used in the spoken Arabic dialect used in some countries along the shores of the Persian Gulf, which means means *the next*. In standard Arabic, a significantly different form is used.

We can also see in sentence pair 9 that our aligner does not always make the correct choice when determining spans. In this case, the segment *RT @MARYAMALKHAWAJA:* was included in the English segment spuriously, since it does not correspond to anything in the Arabic counterpart.

## 5.2 Machine Translation Experiments

We report on machine translation experiments using our harvested data in two domains: edited news and microblogs.

**News translation.** For the news test, we created a new test set from a crawl of the Chinese-English documents on the Project Syndicate website<sup>2</sup>, which contains news commentary articles. We chose to use this data set, rather than more standard NIST test sets to ensure that we had recent documents in the test set (the most recent NIST test sets contain documents published in 2007, well before our microblog data was created). We extracted 1386 parallel sentences for tuning and another 1386 sentences for testing, from the manually aligned segments. For this test set, we used 8 million sentences from the full NIST parallel dataset as the language model training data. We shall call this test set **Syndicate**.

<sup>2</sup><http://www.project-syndicate.org/>

	ENGLISH	MANDARIN
1	i <b>wanna</b> live in a wes anderson world	我想要生活在Wes Anderson的世界里
2	Chicken soup, corn never truly digests. <b>TMI.</b>	鸡汤吧，玉米神马的从来没有真正消化过. 恶心
3	To DanielVeuleman <b>yea iknw imma</b> work on that	对DanielVeuleman说，是的我知道，我正在向那方面努力
4	<b>msg 4</b> Warren G his <b>cd</b> ay is today 1 <b>yr</b> older.	发信息给Warren G，今天是他的生日，又老了一岁了。
5	Where <b>the hell</b> have you been all these years?	这些年你 <b>TMD</b> 到哪去了
	ENGLISH	ARABIC
6	It's <b>gonna</b> be a warm week!	الاسبوع الياي حر
7	onni this gift only <b>4 u</b>	أوني هذة الهدية فقط لك
8	sunset in aqaba :)	غروب الشمس في العقبة:)
9	RT @MARYAMALKHAWAJA: there is a call for widespread protests in #bahrain <b>tmrw</b>	هناك نداء لمظاهرات في عدة مناطق غدا

Table 2: Examples of English-Mandarin and English-Arabic sentence pairs. The English-Mandarin sentences were extracted from Sina Weibo and the English-Arabic sentences were extracted from Twitter. Some messages have been shorted to fit into the table. Some interesting aspects of these sentence pairs are marked in bold.

**Microblog translation.** To carry out the microblog translation experiments, we need a high quality parallel test set. Since we are not aware of such a test set, we created one by manually selecting parallel messages from Weibo. Our procedure was as follows. We selected 2000 candidate Weibo posts from users who have a high number of parallel tweets according to our automatic method (at least 2 in every 5 tweets). To these, we added another 2000 messages from our targeted Weibo crawl, but these had no requirement on the proportion of parallel tweets they had produced. We identified 2374 parallel segments, of which we used 1187 for development and 1187 for testing. We refer to this test set as **Weibo**.<sup>3</sup>

Obviously, we removed the development and test sets from our training data. Furthermore, to ensure that our training data was not too similar to the test set in the Weibo translation task, we filtered the *training* data to remove near duplicates by computing edit distance between each parallel sentence in the heldout set and each training instance. If either the source or the target sides of the a training instance had an edit distance of less than 10%, we removed it.<sup>4</sup> As for the language models, we collected a further 10M tweets from Twitter for the English language model and another 10M tweets from Weibo for the Chinese language model.

<sup>3</sup>We acknowledge that self-translated messages are probably not a typically representative sample of all microblog messages. However, we do not have the resources to produce a carefully curated test set with a more broadly representative distribution. Still, we believe these results are informative as long as this is kept in mind.

<sup>4</sup>Approximately 150,000 training instances removed.

	Syndicate		Weibo	
	ZH-EN	EN-ZH	ZH-EN	EN-ZH
FBIS	9.4	18.6	10.4	12.3
NIST	11.5	21.2	11.4	13.9
Weibo	8.75	15.9	15.7	17.2
FBIS+Weibo	11.7	19.2	16.5	17.8
NIST+Weibo	<b>13.3</b>	<b>21.5</b>	<b>16.9</b>	<b>17.9</b>

Table 3: BLEU scores for different datasets in different translation directions (left to right), broken with different training corpora (top to bottom).

**Baselines.** We report results on these test sets using different training data. First, we use the FBIS dataset which contains 300K high quality sentence pairs, mostly in the broadcast news domain. Second, we use the full 2012 NIST Chinese-English dataset (approximately 8M sentence pairs, including FBIS). Finally, we use our crawled data (referred as Weibo) by itself and also combined with the two previous training sets.

**Setup.** We use the Moses phrase-based MT system with standard features (Koehn et al., 2003). For reordering, we use the MSD reordering model (Axelrod et al., 2005). As the language model, we use a 5-gram model with Kneser-Ney smoothing. The weights were tuned using MERT (Och, 2003). Results are presented with BLEU-4 (Papineni et al., 2002).

**Results.** The BLEU scores for the different parallel corpora are shown in Table 3 and the top 10 out-of-vocabulary (OOV) words for each dataset are shown in Table 4. We observe that for the **Syndicate** test set, the NIST and FBIS datasets

Syndicate (test)			Weibo (test)		
FBIS	NIST	Weibo	FBIS	NIST	Weibo
obama (83)	barack (59)	democracies (15)	2012 (24)	showstudio (9)	submissions (4)
barack (59)	namo (6)	imbalances (13)	alanis (13)	crue (9)	ivillage (4)
princeton (40)	mitt (6)	mahmoud (12)	crue (9)	overexposed (8)	scola (3)
ecb (8)	guant (6)	millennium (9)	showstudio (9)	tweetmeian (5)	rbst (3)
bernanke (8)	fairtrade (6)	regimes (8)	overexposed (8)	tvd (5)	curitiba (3)
romney (7)	hollande (5)	wolfowitz (7)	itunes (8)	iheartradio (5)	zeman (2)
gaddafi (7)	wikileaks (4)	revolutions (7)	havoc (8)	xoxo (4)	@yaptv (2)
merkel (7)	wilders (3)	qaddafi (7)	sammy (6)	snoop (4)	witnessing (2)
fats (7)	rant (3)	geopolitical (7)	obama (6)	shinoda (4)	whoohooo (2)
dialogue (7)	esm (3)	genome (7)	lol (6)	scrapbook (4)	wbr (2)

Table 4: The most frequent out-of-vocabulary (OOV) words and their counts for the two English-source test sets with three different training sets.

perform better than our extracted parallel data. This is to be expected, since our dataset was extracted from an extremely different domain. However, by combining the Weibo parallel data with this standard data, improvements in BLEU are obtained. Error analysis indicates that one major factor is that names from current events, such as *Romney* and *Wikileaks* do not occur in the older NIST and FBIS datasets, but they are represented in the Weibo dataset. Furthermore, we also note that the system built on the Weibo dataset does not perform substantially worse than the one trained on the FBIS dataset, a further indication that harvesting parallel microblog data yields a diverse collection of translated material.

For the **Weibo** test set, a significant improvement over the news datasets can be achieved using our crawled parallel data. Once again newer terms, such as *iTunes*, are one of the reasons older datasets perform less well. However, in this case, the top OOV words of the news domain datasets are not the most accurate representation of coverage problems in this domain. This is because many frequent words in microblogs, e.g., nonstandard abbreviations, like *u* and *4* are found in the news domain as words, albeit with different meanings. Thus, the OOV table gives an incomplete picture of the translation problems when using the news domain corpora to translate microblogs. Also, some structural errors occur when training with the news domain datasets, one such example is shown in table 5, where the character 说 is incorrectly translated to *said*. This occurs because this type of constructions is infrequent in news datasets. Furthermore, we can see that compound expressions, such as the translation from 派对时刻 to *party time* are also learned.

Finally, we observe that combining the datasets

Source	对sam farrar 说, 派对时刻
Reference	to sam farrar , party time
FBIS	farrar to sam said , in time
NIST	to sam farrar said , the moment
WEIBO	to sam farrar , party time

Table 5: Translation Examples using different training sets.

yields another gain over individual datasets, both in the **Syndicate** and in the **Weibo** test sets.

## 6 Conclusion

We presented a framework to crawl parallel data from microblogs. We find parallel data from single posts, with translations of the same sentence in two languages. We show that a considerable amount of parallel sentence pairs can be crawled from microblogs and these can be used to improve Machine Translation by updating our translation tables with translations of newer terms. Furthermore, the in-domain data can substantially improve the translation quality on microblog data.

The resources described in this paper and further developments are available to the general public at <http://www.cs.cmu.edu/~lingwang/utopia>.

## Acknowledgements

The PhD thesis of Wang Ling is supported by FCT grant SFRH/BD/51157/2010. The authors wish to express their gratitude to thank William Cohen, Noah Smith, Waleed Ammar, and the anonymous reviewers for their insight and comments. We are also extremely grateful to Brendan O’Connor for providing the Twitter data and to Philipp Koehn and Barry Haddow for providing the Project Syndicate data.

## References

- [Axelrod et al.2005] Amittai Axelrod, Ra Birch Mayne, Chris Callison-burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*.
- [Blei et al.2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.
- [Braune and Fraser2010] Fabienne Braune and Alexander Fraser. 2010. Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 81–89, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [Brown et al.1993] Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19:263–311, June.
- [Fukushima et al.2006] Ken'ichi Fukushima, Kenjiro Taura, and Takashi Chikayama. 2006. A fast and accurate method for detecting English-Japanese parallel texts. In *Proceedings of the Workshop on Multilingual Language Resources and Interoperability*, pages 60–67, Sydney, Australia, July. Association for Computational Linguistics.
- [Gimpel et al.2011] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [Jelh et al.2012] Laura Jelh, Felix Hiebel, and Stefan Riezler. 2012. Twitter translation using translation-based cross-lingual retrieval. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 410–421, Montréal, Canada, June. Association for Computational Linguistics.
- [Koehn et al.2003] Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54, Morristown, NJ, USA. Association for Computational Linguistics.
- [Koehn2005] Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.
- [Li and Liu2008] Bo Li and Juan Liu. 2008. Mining Chinese-English parallel corpora from the web. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP)*.
- [Lin et al.2008] Dekang Lin, Shaojun Zhao, Benjamin Van Durme, and Marius Paşca. 2008. Mining par-enthetical translations from the web by word alignment. In *Proceedings of ACL-08: HLT*, pages 994–1002, Columbus, Ohio, June. Association for Computational Linguistics.
- [Och2003] Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [Papineni et al.2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [Post et al.2012] Matt Post, Chris Callison-Burch, and Miles Osborne. 2012. Constructing parallel corpora for six indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, Montréal, Canada, June. Association for Computational Linguistics.
- [Resnik and Smith2003] Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29:349–380.
- [Smith et al.2010] Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.
- [Ture and Lin2012] Ferhan Ture and Jimmy Lin. 2012. Why not grab a free lunch? mining large corpora for parallel sentences to improve translation modeling. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 626–630, Montréal, Canada, June. Association for Computational Linguistics.
- [Uszkoreit et al.2010] Jakob Uszkoreit, Jay Ponte, Ashok C. Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1101–1109.
- [Vogel et al.1996] Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics - Volume 2*, COLING '96, pages 836–841, Stroudsburg, PA, USA. Association for Computational Linguistics.
- [Xu et al.2001] Jinxi Xu, Ralph Weischedel, and Chanh Nguyen. 2001. Evaluating a probabilistic model

for cross-lingual information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 105–110, New York, NY, USA. ACM.

[Xu et al.2005] Jia Xu, Richard Zens, and Hermann Ney. 2005. Sentence segmentation using ibm word alignment model 1. In *Proceedings of EAMT 2005 (10th Annual Conference of the European Association for Machine Translation)*, pages 280–287.

[Zbib et al.2012] Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwarz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. 2012. Machine translation of Arabic dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

# Improved Bayesian Logistic Supervised Topic Models with Data Augmentation

Jun Zhu, Xun Zheng, Bo Zhang

Department of Computer Science and Technology

TNLIST Lab and State Key Lab of Intelligent Technology and Systems

Tsinghua University, Beijing, China

{dcszj, dcszb}@tsinghua.edu.cn; vforveri.zheng@gmail.com

## Abstract

Supervised topic models with a logistic likelihood have two issues that potentially limit their practical use: 1) response variables are usually over-weighted by document word counts; and 2) existing variational inference methods make strict mean-field assumptions. We address these issues by: 1) introducing a regularization constant to better balance the two parts based on an optimization formulation of Bayesian inference; and 2) developing a simple Gibbs sampling algorithm by introducing auxiliary Polya-Gamma variables and collapsing out Dirichlet variables. Our augment-and-collapse sampling algorithm has analytical forms of each conditional distribution without making any restricting assumptions and can be easily parallelized. Empirical results demonstrate significant improvements on prediction performance and time efficiency.

## 1 Introduction

As widely adopted in supervised latent Dirichlet allocation (sLDA) models (Blei and McAuliffe, 2010; Wang et al., 2009), one way to improve the predictive power of LDA is to define a likelihood model for the widely available document-level response variables, in addition to the likelihood model for document words. For example, the logistic likelihood model is commonly used for binary or multinomial responses. By imposing some priors, posterior inference is done with the Bayes' rule. Though powerful, one issue that could limit the use of existing logistic supervised LDA models is that they treat the document-level response variable as one additional word via a normalized likelihood model. Although some special treatment is carried out on defining the likelihood of the single

response variable, it is normally of a much smaller scale than the likelihood of the usually tens or hundreds of words in each document. As noted by (Halpern et al., 2012) and observed in our experiments, this model imbalance could result in a weak influence of response variables on the topic representations and thus non-satisfactory prediction performance. Another difficulty arises when dealing with categorical response variables is that the commonly used normal priors are no longer conjugate to the logistic likelihood and thus lead to hard inference problems. Existing approaches rely on variational approximation techniques which normally make strict mean-field assumptions.

To address the above issues, we present two improvements. First, we present a general framework of Bayesian logistic supervised topic models with a regularization parameter to better balance response variables and words. Technically, instead of doing standard Bayesian inference via Bayes' rule, which requires a normalized likelihood model, we propose to do regularized Bayesian inference (Zhu et al., 2011; Zhu et al., 2013b) via solving an optimization problem, where the posterior regularization is defined as an expectation of a logistic loss, a surrogate loss of the expected misclassification error; and a regularization parameter is introduced to balance the surrogate classification loss (i.e., the response log-likelihood) and the word likelihood. The general formulation subsumes standard sLDA as a special case.

Second, to solve the intractable posterior inference problem of the generalized Bayesian logistic supervised topic models, we present a simple Gibbs sampling algorithm by exploring the ideas of data augmentation (Tanner and Wong, 1987; van Dyk and Meng, 2001; Holmes and Held, 2006). More specifically, we extend Polson's method for Bayesian logistic regression (Polson et al., 2012) to the generalized logistic supervised topic models, which are much more challeng-

ing due to the presence of non-trivial latent variables. Technically, we introduce a set of Polya-Gamma variables, one per document, to reformulate the generalized logistic pseudo-likelihood model (with the regularization parameter) as a scale mixture, where the mixture component is conditionally normal for classifier parameters. Then, we develop a simple and efficient Gibbs sampling algorithms with analytic conditional distributions without Metropolis-Hastings accept/reject steps. For Bayesian LDA models, we can also explore the conjugacy of the Dirichlet-Multinomial prior-likelihood pairs to collapse out the Dirichlet variables (i.e., topics and mixing proportions) to do collapsed Gibbs sampling, which can have better mixing rates (Griffiths and Steyvers, 2004). Finally, our empirical results on real data sets demonstrate significant improvements on time efficiency. The classification performance is also significantly improved by using appropriate regularization parameters. We also provide a parallel implementation with GraphLab (Gonzalez et al., 2012), which shows great promise in our preliminary studies.

The paper is structured as follows. Sec. 2 introduces logistic supervised topic models as a general optimization problem. Sec. 3 presents Gibbs sampling algorithms with data augmentation. Sec. 4 presents experiments. Sec. 5 concludes.

## 2 Logistic Supervised Topic Models

We now present the generalized Bayesian logistic supervised topic models.

### 2.1 The Generalized Models

We consider binary classification with a training set  $\mathcal{D} = \{(\mathbf{w}_d, y_d)\}_{d=1}^D$ , where the response variable  $Y$  takes values from the output space  $\mathcal{Y} = \{0, 1\}$ . A logistic supervised topic model consists of two parts — an LDA model (Blei et al., 2003) for describing the words  $\mathbf{W} = \{\mathbf{w}_d\}_{d=1}^D$ , where  $\mathbf{w}_d = \{w_{dn}\}_{n=1}^{N_d}$  denote the words within document  $d$ , and a logistic classifier for considering the supervising signal  $\mathbf{y} = \{y_d\}_{d=1}^D$ . Below, we introduce each of them in turn.

**LDA:** LDA is a hierarchical Bayesian model that posits each document as an admixture of  $K$  topics, where each topic  $\Phi_k$  is a multinomial distribution over a  $V$ -word vocabulary. For document  $d$ , the generating process is

1. draw a topic proportion  $\theta_d \sim \text{Dir}(\boldsymbol{\alpha})$
2. for each word  $n = 1, 2, \dots, N_d$ :

- (a) draw a topic<sup>1</sup>  $z_{dn} \sim \text{Mult}(\boldsymbol{\theta}_d)$
- (b) draw the word  $w_{dn} \sim \text{Mult}(\Phi_{z_{dn}})$

where  $\text{Dir}(\cdot)$  is a Dirichlet distribution;  $\text{Mult}(\cdot)$  is a multinomial distribution; and  $\Phi_{z_{dn}}$  denotes the topic selected by the non-zero entry of  $z_{dn}$ . For fully-Bayesian LDA, the topics are random samples from a Dirichlet prior,  $\Phi_k \sim \text{Dir}(\boldsymbol{\beta})$ .

Let  $\mathbf{z}_d = \{z_{dn}\}_{n=1}^{N_d}$  denote the set of topic assignments for document  $d$ . Let  $\mathbf{Z} = \{\mathbf{z}_d\}_{d=1}^D$  and  $\Theta = \{\theta_d\}_{d=1}^D$  denote all the topic assignments and mixing proportions for the entire corpus. LDA infers the posterior distribution  $p(\Theta, \mathbf{Z}, \Phi | \mathbf{W}) \propto p_0(\Theta, \mathbf{Z}, \Phi) p(\mathbf{W} | \mathbf{Z}, \Phi)$ , where  $p_0(\Theta, \mathbf{Z}, \Phi) = (\prod_d p(\theta_d | \boldsymbol{\alpha}) \prod_n p(z_{dn} | \theta_d)) \prod_k p(\Phi_k | \boldsymbol{\beta})$  is the joint distribution defined by the model. As noticed in (Jiang et al., 2012), the posterior distribution by Bayes' rule is equivalent to the solution of an information theoretical optimization problem

$$\min_{q(\Theta, \mathbf{Z}, \Phi)} \text{KL}(q(\Theta, \mathbf{Z}, \Phi) \| p_0(\Theta, \mathbf{Z}, \Phi)) - \mathbb{E}_q[\log p(\mathbf{W} | \mathbf{Z}, \Phi)]$$

s.t. :  $q(\Theta, \mathbf{Z}, \Phi) \in \mathcal{P}$ ,

(1)

where  $\text{KL}(q \| p)$  is the Kullback-Leibler divergence from  $q$  to  $p$  and  $\mathcal{P}$  is the space of probability distributions.

**Logistic classifier:** To consider binary supervising information, a logistic supervised topic model (e.g., sLDA) builds a logistic classifier using the topic representations as input features

$$p(y = 1 | \boldsymbol{\eta}, \mathbf{z}) = \frac{\exp(\boldsymbol{\eta}^\top \bar{\mathbf{z}})}{1 + \exp(\boldsymbol{\eta}^\top \bar{\mathbf{z}})},$$
(2)

where  $\bar{\mathbf{z}}$  is a  $K$ -vector with  $\bar{z}_k = \frac{1}{N} \sum_{n=1}^N \mathbb{I}(z_n^k = 1)$ , and  $\mathbb{I}(\cdot)$  is an indicator function that equals to 1 if predicate holds otherwise 0. If the classifier weights  $\boldsymbol{\eta}$  and topic assignments  $\mathbf{z}$  are given, the prediction rule is

$$\hat{y} |_{\boldsymbol{\eta}, \mathbf{z}} = \mathbb{I}(p(y = 1 | \boldsymbol{\eta}, \mathbf{z}) > 0.5) = \mathbb{I}(\boldsymbol{\eta}^\top \bar{\mathbf{z}} > 0).$$
(3)

Since both  $\boldsymbol{\eta}$  and  $\mathbf{Z}$  are hidden variables, we propose to infer a posterior distribution  $q(\boldsymbol{\eta}, \mathbf{Z})$  that has the minimal expected log-logistic loss

$$\mathcal{R}(q(\boldsymbol{\eta}, \mathbf{Z})) = - \sum_d \mathbb{E}_q[\log p(y_d | \boldsymbol{\eta}, \mathbf{z}_d)],$$
(4)

which is a good surrogate loss for the expected misclassification loss,  $\sum_d \mathbb{E}_q[\mathbb{I}(\hat{y} |_{\boldsymbol{\eta}, \mathbf{z}_d} \neq y_d)]$ , of a Gibbs classifier that randomly draws a model  $\boldsymbol{\eta}$  from the posterior distribution and makes predictions (McAllester, 2003; Germain et al., 2009). In fact, this choice is motivated from the observation that logistic loss has been widely used as a convex surrogate loss for the misclassification

<sup>1</sup>A  $K$ -binary vector with only one entry equaling to 1.

loss (Rosasco et al., 2004) in the task of fully observed binary classification. Also, note that the logistic classifier and the LDA likelihood are coupled by sharing the latent topic assignments  $\mathbf{z}$ . The strong coupling makes it possible to learn a posterior distribution that can describe the observed words well and make accurate predictions.

**Regularized Bayesian Inference:** To integrate the above two components for hybrid learning, a logistic supervised topic model solves the joint Bayesian inference problem

$$\begin{aligned} \min_{q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})} \mathcal{L}(q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})) + c\mathcal{R}(q(\boldsymbol{\eta}, \mathbf{Z})) \quad (5) \\ \text{s.t.: } q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) \in \mathcal{P}, \end{aligned}$$

where  $\mathcal{L}(q) = \text{KL}(q||p_0(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})) - \mathbb{E}_q[\log p(\mathbf{W}|\mathbf{Z}, \boldsymbol{\Phi})]$  is the objective for doing standard Bayesian inference with the classifier weights  $\boldsymbol{\eta}$ ;  $p_0(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) = p_0(\boldsymbol{\eta})p_0(\boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})$ ; and  $c$  is a regularization parameter balancing the influence from response variables and words.

In general, we define the pseudo-likelihood for the supervision information

$$\psi(y_d|\mathbf{z}_d, \boldsymbol{\eta}) = p^c(y_d|\boldsymbol{\eta}, \mathbf{z}_d) = \frac{\{\exp(\boldsymbol{\eta}^\top \bar{\mathbf{z}}_d)\}^{cy_d}}{(1 + \exp(\boldsymbol{\eta}^\top \bar{\mathbf{z}}_d))^c}, \quad (6)$$

which is un-normalized if  $c \neq 1$ . But, as we shall see this un-normalization does not affect our subsequent inference. Then, the generalized inference problem (5) of logistic supervised topic models can be written in the ‘‘standard’’ Bayesian inference form (1)

$$\begin{aligned} \min_{q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})} \mathcal{L}(q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})) - \mathbb{E}_q[\log \psi(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta})] \quad (7) \\ \text{s.t.: } q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) \in \mathcal{P}, \end{aligned}$$

where  $\psi(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta}) = \prod_d \psi(y_d|\mathbf{z}_d, \boldsymbol{\eta})$ . It is easy to show that the optimum solution of problem (5) or the equivalent problem (7) is the posterior distribution with supervising information, i.e.,

$$q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) = \frac{p_0(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})p(\mathbf{W}|\mathbf{Z}, \boldsymbol{\Phi})\psi(\mathbf{y}|\boldsymbol{\eta}, \mathbf{Z})}{\phi(\mathbf{y}, \mathbf{W})}.$$

where  $\phi(\mathbf{y}, \mathbf{W})$  is the normalization constant to make  $q$  a distribution. We can see that when  $c = 1$ , the model reduces to the standard sLDA, which in practice has the imbalance issue that the response variable (can be viewed as one additional word) is usually dominated by the words. This imbalance was noticed in (Halpern et al., 2012). We will see that  $c$  can make a big difference later.

**Comparison with MedLDA:** The above formulation of logistic supervised topic models as an instance of regularized Bayesian inference provides a direct comparison with the max-margin

supervised topic model (MedLDA) (Jiang et al., 2012), which has the same form of the optimization problems. The difference lies in the posterior regularization, for which MedLDA uses a hinge loss of an expected classifier while the logistic supervised topic model uses an expected log-logistic loss. Gibbs MedLDA (Zhu et al., 2013a) is another max-margin model that adopts the expected hinge loss as posterior regularization. As we shall see in the experiments, by using appropriate regularization constants, logistic supervised topic models achieve comparable performance as max-margin methods. We note that the relationship between a logistic loss and a hinge loss has been discussed extensively in various settings (Rosasco et al., 2004; Globerson et al., 2007). But the presence of latent variables poses additional challenges in carrying out a formal theoretical analysis of these surrogate losses (Lin, 2001) in the topic model setting.

## 2.2 Variational Approximation Algorithms

The commonly used normal prior for  $\boldsymbol{\eta}$  is non-conjugate to the logistic likelihood, which makes the posterior inference hard. Moreover, the latent variables  $\mathbf{Z}$  make the inference problem harder than that of Bayesian logistic regression models (Chen et al., 1999; Meyer and Laud, 2002; Polson et al., 2012). Previous algorithms to solve problem (5) rely on variational approximation techniques. It is easy to show that the variational method (Wang et al., 2009) is a coordinate descent algorithm to solve problem (5) with the additional fully-factorized constraint  $q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) = q(\boldsymbol{\eta})(\prod_d q(\boldsymbol{\theta}_d) \prod_n q(z_{dn})) \prod_k q(\boldsymbol{\Phi}_k)$  and a variational approximation to the expectation of the log-logistic likelihood, which is intractable to compute directly. Note that the non-Bayesian treatment of  $\boldsymbol{\eta}$  as unknown parameters in (Wang et al., 2009) results in an EM algorithm, which still needs to make strict mean-field assumptions together with a variational bound of the expectation of the log-logistic likelihood. In this paper, we consider the full Bayesian treatment, which can principally consider prior distributions and infer the posterior covariance.

## 3 A Gibbs Sampling Algorithm

Now, we present a simple and efficient Gibbs sampling algorithm for the generalized Bayesian logistic supervised topic models.

### 3.1 Formulation with Data Augmentation

Since the logistic pseudo-likelihood  $\psi(\mathbf{y}|\mathbf{Z}, \boldsymbol{\eta})$  is not conjugate with normal priors, it is not easy to derive the sampling algorithms directly. Instead, we develop our algorithms by introducing auxiliary variables, which lead to a scale mixture of Gaussian components and analytic conditional distributions for automatical Bayesian inference without an accept/reject ratio. Our algorithm represents a first attempt to extend Polson's approach (Polson et al., 2012) to deal with highly non-trivial Bayesian latent variable models. Let us first introduce the Polya-Gamma variables.

**Definition 1** (Polson et al., 2012) *A random variable  $X$  has a Polya-Gamma distribution, denoted by  $X \sim \mathcal{PG}(a, b)$ , if*

$$X = \frac{1}{2\pi^2} \sum_{i=1}^{\infty} \frac{g_k}{(i-1)^2/2 + b^2/(4\pi^2)},$$

where  $a, b > 0$  and each  $g_i \sim \mathcal{G}(a, 1)$  is an independent Gamma random variable.

Let  $\omega_d = \boldsymbol{\eta}^\top \bar{\mathbf{z}}_d$ . Then, using the ideas of data augmentation (Tanner and Wong, 1987; Polson et al., 2012), we can show that the generalized pseudo-likelihood can be expressed as

$$\psi(y_d|\mathbf{z}_d, \boldsymbol{\eta}) = \frac{1}{2^c} e^{\kappa_d \omega_d} \int_0^\infty \exp\left(-\frac{\lambda_d \omega_d^2}{2}\right) p(\lambda_d|c, 0) d\lambda_d,$$

where  $\kappa_d = c(y_d - 1/2)$  and  $\lambda_d$  is a Polya-Gamma variable with parameters  $a = c$  and  $b = 0$ . This result indicates that the posterior distribution of the generalized Bayesian logistic supervised topic models, i.e.,  $q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})$ , can be expressed as the marginal of a higher dimensional distribution that includes the augmented variables  $\boldsymbol{\lambda}$ . The complete posterior distribution is

$$q(\boldsymbol{\eta}, \boldsymbol{\lambda}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) = \frac{p_0(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi}) p(\mathbf{W}|\mathbf{Z}, \boldsymbol{\Phi}) \phi(\mathbf{y}, \boldsymbol{\lambda}|\mathbf{Z}, \boldsymbol{\eta})}{\psi(\mathbf{y}, \mathbf{W})},$$

where the pseudo-joint distribution of  $\mathbf{y}$  and  $\boldsymbol{\lambda}$  is

$$\phi(\mathbf{y}, \boldsymbol{\lambda}|\mathbf{Z}, \boldsymbol{\eta}) = \prod_d \exp\left(\kappa_d \omega_d - \frac{\lambda_d \omega_d^2}{2}\right) p(\lambda_d|c, 0).$$

### 3.2 Inference with Collapsed Gibbs Sampling

Although we can do Gibbs sampling to infer the complete posterior distribution  $q(\boldsymbol{\eta}, \boldsymbol{\lambda}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})$  and thus  $q(\boldsymbol{\eta}, \boldsymbol{\Theta}, \mathbf{Z}, \boldsymbol{\Phi})$  by ignoring  $\boldsymbol{\lambda}$ , the mixing rate would be slow due to the large sample space. One way to effectively improve mixing rates is to integrate out the intermediate variables  $(\boldsymbol{\Theta}, \boldsymbol{\Phi})$  and build a Markov chain whose equilibrium distribution is the marginal distribution  $q(\boldsymbol{\eta}, \boldsymbol{\lambda}, \mathbf{Z})$ . We propose to use collapsed Gibbs

sampling, which has been successfully used in LDA (Griffiths and Steyvers, 2004). For our model, the collapsed posterior distribution is

$$\begin{aligned} q(\boldsymbol{\eta}, \boldsymbol{\lambda}, \mathbf{Z}) &\propto p_0(\boldsymbol{\eta}) p(\mathbf{W}, \mathbf{Z}|\boldsymbol{\alpha}, \boldsymbol{\beta}) \phi(\mathbf{y}, \boldsymbol{\lambda}|\mathbf{Z}, \boldsymbol{\eta}) \\ &= p_0(\boldsymbol{\eta}) \prod_{k=1}^K \frac{\delta(\mathbf{C}_k + \boldsymbol{\beta})}{\delta(\boldsymbol{\beta})} \prod_{d=1}^D \left[ \frac{\delta(\mathbf{C}_d + \boldsymbol{\alpha})}{\delta(\boldsymbol{\alpha})} \right. \\ &\quad \left. \times \exp\left(\kappa_d \omega_d - \frac{\lambda_d \omega_d^2}{2}\right) p(\lambda_d|c, 0) \right], \end{aligned}$$

where  $\delta(\mathbf{x}) = \frac{\prod_{i=1}^{\dim(\mathbf{x})} \Gamma(x_i)}{\Gamma(\sum_{i=1}^{\dim(\mathbf{x})} x_i)}$ ,  $C_k^t$  is the number of times the term  $t$  being assigned to topic  $k$  over the whole corpus and  $\mathbf{C}_k = \{C_k^t\}_{t=1}^V$ ;  $C_d^k$  is the number of times that terms being associated with topic  $k$  within the  $d$ -th document and  $\mathbf{C}_d = \{C_d^k\}_{k=1}^K$ . Then, the conditional distributions used in collapsed Gibbs sampling are as follows.

**For  $\boldsymbol{\eta}$ :** for the commonly used isotropic Gaussian prior  $p_0(\boldsymbol{\eta}) = \prod_k \mathcal{N}(\eta_k; 0, \nu^2)$ , we have

$$\begin{aligned} q(\boldsymbol{\eta}|\mathbf{Z}, \boldsymbol{\lambda}) &\propto p_0(\boldsymbol{\eta}) \prod_d \exp\left(\kappa_d \omega_d - \frac{\lambda_d \omega_d^2}{2}\right) \\ &= \mathcal{N}(\boldsymbol{\eta}; \boldsymbol{\mu}, \Sigma), \end{aligned} \quad (8)$$

where the posterior mean is  $\boldsymbol{\mu} = \Sigma(\sum_d \kappa_d \bar{\mathbf{z}}_d)$  and the covariance is  $\Sigma = (\frac{1}{\nu^2} I + \sum_d \lambda_d \bar{\mathbf{z}}_d \bar{\mathbf{z}}_d^\top)^{-1}$ . We can easily draw a sample from a  $K$ -dimensional multivariate Gaussian distribution. The inverse can be robustly done using Cholesky decomposition, an  $O(K^3)$  procedure. Since  $K$  is normally not large, the inversion can be done efficiently.

**For  $\mathbf{Z}$ :** The conditional distribution of  $\mathbf{Z}$  is

$$\begin{aligned} q(\mathbf{Z}|\boldsymbol{\eta}, \boldsymbol{\lambda}) &\propto \prod_{k=1}^K \frac{\delta(\mathbf{C}_k + \boldsymbol{\beta})}{\delta(\boldsymbol{\beta})} \prod_{d=1}^D \left[ \frac{\delta(\mathbf{C}_d + \boldsymbol{\alpha})}{\delta(\boldsymbol{\alpha})} \right. \\ &\quad \left. \times \exp\left(\kappa_d \omega_d - \frac{\lambda_d \omega_d^2}{2}\right) \right]. \end{aligned}$$

By canceling common factors, we can derive the local conditional of one variable  $z_{dn}$  as:

$$\begin{aligned} q(z_{dn}^k = 1 | \mathbf{Z}_{-n}, \boldsymbol{\eta}, \boldsymbol{\lambda}, w_{dn} = t) &\propto \frac{(C_{k,-n}^t + \beta_t)(C_{d,-n}^k + \alpha_k)}{\sum_t C_{k,-n}^t + \sum_{t=1}^V \beta_t} \exp\left(\gamma \kappa_d \eta_k \right. \\ &\quad \left. - \lambda_d \frac{\gamma^2 \eta_k^2 + 2\gamma(1-\gamma)\eta_k \Lambda_{dn}^k}{2}\right), \end{aligned} \quad (9)$$

where  $C_{\cdot,-n}$  indicates that term  $n$  is excluded from the corresponding document or topic;  $\gamma = \frac{1}{N_d}$ ; and  $\Lambda_{dn}^k = \frac{1}{N_d - 1} \sum_{k'} \eta_{k'} C_{d,-n}^{k'}$  is the discriminant function value without word  $n$ . We can see that the first term is from the LDA model for observed word counts and the second term is from the supervising signal  $\mathbf{y}$ .

**For  $\boldsymbol{\lambda}$ :** Finally, the conditional distribution of the augmented variables  $\boldsymbol{\lambda}$  is

$$\begin{aligned} q(\lambda_d|\mathbf{Z}, \boldsymbol{\eta}) &\propto \exp\left(-\frac{\lambda_d \omega_d^2}{2}\right) p(\lambda_d|c, 0) \\ &= \mathcal{PG}(\lambda_d; c, \omega_d), \end{aligned} \quad (10)$$

---

**Algorithm 1** for collapsed Gibbs sampling

---

- 1: **Initialization:** set  $\lambda = 1$  and randomly draw  $z_{dn}$  from a uniform distribution.
  - 2: **for**  $m = 1$  **to**  $M$  **do**
  - 3:   draw a classifier from the distribution (8)
  - 4:   **for**  $d = 1$  **to**  $D$  **do**
  - 5:     **for** each word  $n$  in document  $d$  **do**
  - 6:       draw the topic using distribution (9)
  - 7:     **end for**
  - 8:     draw  $\lambda_d$  from distribution (10).
  - 9:   **end for**
  - 10: **end for**
- 

which is a Polya-Gamma distribution. The equality has been achieved by using the construction definition of the general  $\mathcal{PG}(a, b)$  class through an exponential tilting of the  $\mathcal{PG}(a, 0)$  density (Polson et al., 2012). To draw samples from the Polya-Gamma distribution, we adopt the efficient method<sup>2</sup> proposed in (Polson et al., 2012), which draws the samples through drawing samples from the closely related exponentially tilted Jacobi distribution.

With the above conditional distributions, we can construct a Markov chain which iteratively draws samples of  $\eta$  using Eq. (8),  $\mathbf{Z}$  using Eq. (9) and  $\lambda$  using Eq. (10), with an initial condition. In our experiments, we initially set  $\lambda = 1$  and randomly draw  $\mathbf{Z}$  from a uniform distribution. In training, we run the Markov chain for  $M$  iterations (i.e., the burn-in stage), as outlined in Algorithm 1. Then, we draw a sample  $\hat{\eta}$  as the final classifier to make predictions on testing data. As we shall see, the Markov chain converges to stable prediction performance with a few burn-in iterations.

### 3.3 Prediction

To apply the classifier  $\hat{\eta}$  on testing data, we need to infer their topic assignments. We take the approach in (Zhu et al., 2012; Jiang et al., 2012), which uses a point estimate of topics  $\Phi$  from training data and makes prediction based on them. Specifically, we use the MAP estimate  $\hat{\Phi}$  to replace the probability distribution  $p(\Phi)$ . For the Gibbs sampler, an estimate of  $\hat{\Phi}$  using the samples is  $\hat{\phi}_{kt} \propto C_k^t + \beta_t$ . Then, given a testing document  $\mathbf{w}$ , we infer its latent components  $\mathbf{z}$  using  $\hat{\Phi}$  as  $p(z_n = k | \mathbf{z}_{-n}) \propto \hat{\phi}_{kw_n} (C_{-n}^k + \alpha_k)$ , where

---

<sup>2</sup>The basic sampler was implemented in the R package BayesLogit. We implemented the sampling algorithm in C++ together with our topic model sampler.

$C_{-n}^k$  is the times that the terms in this document  $\mathbf{w}$  assigned to topic  $k$  with the  $n$ -th term excluded.

## 4 Experiments

We present empirical results and sensitivity analysis to demonstrate the efficiency and prediction performance<sup>3</sup> of the generalized logistic supervised topic models on the 20Newsgroups (20NG) data set, which contains about 20,000 postings within 20 news groups. We follow the same setting as in (Zhu et al., 2012) and remove a standard list of stop words for both binary and multi-class classification. For all the experiments, we use the standard normal prior  $p_0(\eta)$  (i.e.,  $\nu^2 = 1$ ) and the symmetric Dirichlet priors  $\alpha = \frac{\alpha}{K} \mathbf{1}$ ,  $\beta = 0.01 \times \mathbf{1}$ , where  $\mathbf{1}$  is a vector with all entries being 1. For each setting, we report the average performance and the standard deviation with five randomly initialized runs.

### 4.1 Binary classification

Following the same setting in (Lacoste-Jullien et al., 2009; Zhu et al., 2012), the task is to distinguish postings of the newsgroup *alt.atheism* and those of the group *talk.religion.misc*. The training set contains 856 documents and the test set contains 569 documents. We compare the generalized logistic supervised LDA using Gibbs sampling (denoted by gSLDA) with various competitors, including the standard sLDA using variational mean-field methods (denoted by vSLDA) (Wang et al., 2009), the MedLDA model using variational mean-field methods (denoted by vMedLDA) (Zhu et al., 2012), and the MedLDA model using collapsed Gibbs sampling algorithms (denoted by gMedLDA) (Jiang et al., 2012). We also include the unsupervised LDA using collapsed Gibbs sampling as a baseline, denoted by gLDA. For gLDA, we learn a binary linear SVM on its topic representations using SVMlight (Joachims, 1999). The results of DiscLDA (Lacoste-Jullien et al., 2009) and linear SVM on raw bag-of-words features were reported in (Zhu et al., 2012). For gSLDA, we compare two versions – the standard sLDA with  $c = 1$  and the sLDA with a well-tuned  $c$  value. To distinguish, we denote the latter by gSLDA+. We set  $c = 25$  for gSLDA+, and set  $\alpha = 1$  and  $M = 100$  for both gSLDA and gSLDA+. As we shall see, gSLDA is insensitive to  $\alpha$ ,

---

<sup>3</sup>Due to space limit, the topic visualization (similar to that of MedLDA) is deferred to a longer version.

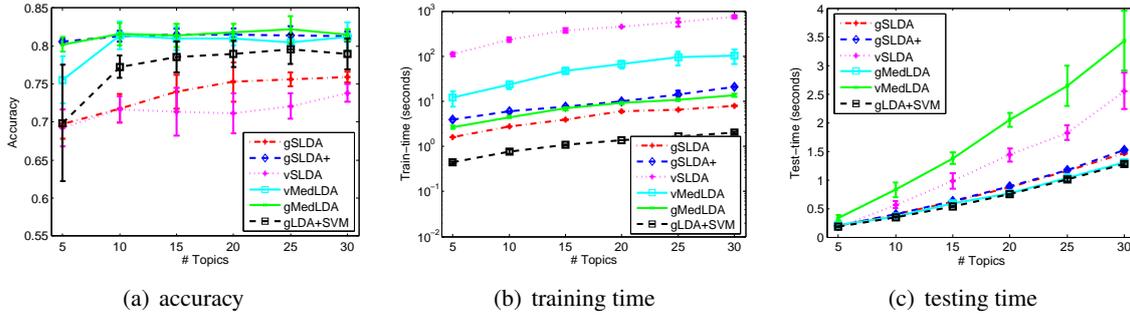


Figure 1: Accuracy, training time (in log-scale) and testing time on the 20NG binary data set.

$c$  and  $M$  in a wide range.

Fig. 1 shows the performance of different methods with various numbers of topics. For accuracy, we can draw two conclusions: 1) without making restricting assumptions on the posterior distributions, gSLDA achieves higher accuracy than vSLDA that uses strict variational mean-field approximation; and 2) by using the regularization constant  $c$  to improve the influence of supervision information, gSLDA+ achieves much better classification results, in fact comparable with those of MedLDA models since they have the similar mechanism to improve the influence of supervision by tuning a regularization constant. The fact that gLDA+SVM performs better than the standard gSLDA is due to the same reason, since the SVM part of gLDA+SVM can well capture the supervision information to learn a classifier for good prediction, while standard sLDA can’t well-balance the influence of supervision. In contrast, the well-balanced gSLDA+ model successfully outperforms the two-stage approach, gLDA+SVM, by performing topic discovery and prediction jointly<sup>4</sup>.

For training time, both gSLDA and gSLDA+ are very efficient, e.g., about 2 orders of magnitudes faster than vSLDA and about 1 order of magnitude faster than vMedLDA. For testing time, gSLDA and gSLDA+ are comparable with gMedLDA and the unsupervised gLDA, but faster than the variational vMedLDA and vSLDA, especially when  $K$  is large.

## 4.2 Multi-class classification

We perform multi-class classification on the 20NG data set with all the 20 categories. For multi-class classification, one possible extension is to use a multinomial logistic regression model for categorical variables  $Y$  by using topic representations  $\bar{z}$  as input features. However, it is non-

<sup>4</sup>The variational sLDA with a well-tuned  $c$  is significantly better than the standard sLDA, but a bit inferior to gSLDA+.

trivial to develop a Gibbs sampling algorithm using the similar data augmentation idea, due to the presence of latent variables and the nonlinearity of the soft-max function. In fact, this is harder than the multinomial Bayesian logistic regression, which can be done via a coordinate strategy (Polson et al., 2012). Here, we apply the binary gSLDA to do the multi-class classification, following the “one-vs-all” strategy, which has been shown effective (Rifkin and Klautau, 2004), to provide some preliminary analysis. Namely, we learn 20 binary gSLDA models and aggregate their predictions by taking the most likely ones as the final predictions. We again evaluate two versions of gSLDA – the standard gSLDA with  $c = 1$  and the improved gSLDA+ with a well-tuned  $c$  value. Since gSLDA is also insensitive to  $\alpha$  and  $c$  for the multi-class task, we set  $\alpha = 5.6$  for both gSLDA and gSLDA+, and set  $c = 256$  for gSLDA+. The number of burn-in is set as  $M = 40$ , which is sufficiently large to get stable results, as we shall see.

Fig. 2 shows the accuracy and training time. We can see that: 1) by using Gibbs sampling without restricting assumptions, gSLDA performs better than the variational vSLDA that uses strict mean-field approximation; 2) due to the imbalance between the single supervision and a large set of word counts, gSLDA doesn’t outperform the decoupled approach, gLDA+SVM; and 3) if we increase the value of the regularization constant  $c$ , supervision information can be better captured to infer predictive topic representations, and gSLDA+ performs much better than gSLDA. In fact, gSLDA+ is even better than the MedLDA that uses mean-field approximation, while is comparable with the MedLDA using collapsed Gibbs sampling. Finally, we should note that the improvement on the accuracy might be due to the different strategies on building the multi-class classifiers. But given the performance gain in the binary task, we believe that the Gibbs sampling algorithm-

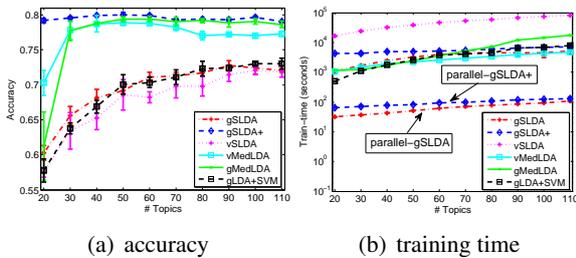


Figure 2: Multi-class classification.

Table 1: Split of training time over various steps.

	SAMPLE $\lambda$	SAMPLE $\eta$	SAMPLE $Z$
K=20	2841.67 (65.80%)	7.70 (0.18%)	1455.25 (34.02%)
K=30	2417.95 (56.10%)	10.34 (0.24%)	1888.78 (43.66%)
K=40	2393.77 (49.00%)	14.66 (0.30%)	2476.82 (50.70%)
K=50	2161.09 (43.67%)	16.33 (0.33%)	2771.26 (56.00%)

m without factorization assumptions is the main factor for the improved performance.

For training time, gSLDA models are about 10 times faster than variational vSLDA. Table 1 shows in detail the percentages of the training time (see the numbers in brackets) spent at each sampling step for gSLDA+. We can see that: 1) sampling the global variables  $\eta$  is very efficient, while sampling local variables ( $\lambda$ ,  $Z$ ) are much more expensive; and 2) sampling  $\lambda$  is relatively stable as  $K$  increases, while sampling  $Z$  takes more time as  $K$  becomes larger. But, the good news is that our Gibbs sampling algorithm can be easily parallelized to speedup the sampling of local variables, following the similar architectures as in LDA.

**A Parallel Implementation:** GraphLab is a graph-based programming framework for parallel computing (Gonzalez et al., 2012). It provides a high-level abstraction of parallel tasks by expressing data dependencies with a distributed graph. GraphLab implements a GAS (gather, apply, scatter) model, where the data required to compute a vertex (edge) are gathered along its neighboring components, and modification of a vertex (edge) will trigger its adjacent components to recompute their values. Since GAS has been successfully applied to several machine learning algorithms<sup>5</sup> including Gibbs sampling of LDA, we choose it as a preliminary attempt to parallelize our Gibbs sampling algorithm. A systematical investigation of the parallel computation with various architectures is interesting, but beyond the scope of this paper.

For our task, since there is no coupling among the 20 binary gSLDA classifiers, we can learn them in parallel. This suggests an efficient hybrid multi-core/multi-machine implementation, which

<sup>5</sup><http://docs.graphlab.org/toolkits.html>

can avoid the time consumption of IPC (i.e., inter-process communication). Namely, we run our experiments on a cluster with 20 nodes where each node is equipped with two 6-core CPUs (2.93GHz). Each node is responsible for learning one binary gSLDA classifier with a parallel implementation on its 12-cores. For each binary gSLDA model, we construct a bipartite graph connecting train documents with corresponding terms. The graph works as follows: 1) the edges contain the token counts and topic assignments; 2) the vertices contain individual topic counts and the augmented variables  $\lambda$ ; 3) the global topic counts and  $\eta$  are aggregated from the vertices periodically, and the topic assignments and  $\lambda$  are sampled asynchronously during the GAS phases. Once started, sampling and signaling will propagate over the graph. One thing to note is that since we cannot directly measure the number of iterations of an asynchronous model, here we estimate it with the total number of topic samplings, which is again aggregated periodically, divided by the number of tokens. We denote the parallel models by parallel-gSLDA ( $c = 1$ ) and parallel-gSLDA+ ( $c = 256$ ). From Fig. 2 (b), we can see that the parallel gSLDA models are about 2 orders of magnitudes faster than their sequential counterpart models, which is very promising. Also, the prediction performance is not sacrificed as we shall see in Fig. 4.

### 4.3 Sensitivity analysis

**Burn-In:** Fig. 3 shows the performance of gSLDA+ with different burn-in steps for binary classification. When  $M = 0$  (see the most left points), the models are built on random topic assignments. We can see that the classification performance increases fast and converges to the stable optimum with about 20 burn-in steps. The training time increases about linearly in general when using more burn-in steps. Moreover, the training time increases linearly as  $K$  increases. In the previous experiments, we set  $M = 100$ .

Fig. 4 shows the performance of gSLDA+ and its parallel implementation (i.e., parallel-gSLDA+) for the multi-class classification with different burn-in steps. We can see when the number of burn-in steps is larger than 20, the performance of gSLDA+ is quite stable. Again, in the log-log scale, since the slopes of the lines in Fig. 4 (b) are close to the constant 1, the training time grows about linearly as the number of

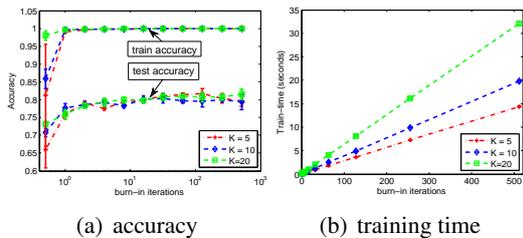


Figure 3: Performance of gSLDA+ with different burn-in steps for binary classification. The most left points are for the settings with no burn in.

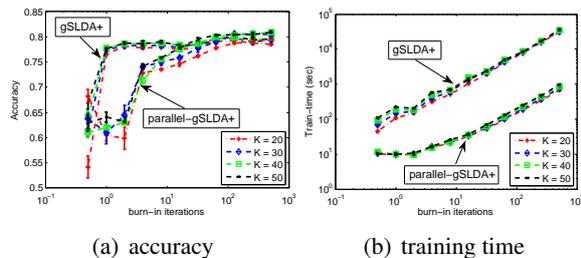


Figure 4: Performance of gSLDA+ and parallel-gSLDA+ with different burn-in steps for multi-class classification. The most left points are for the settings with no burn in.

burn-in steps increases. Even when we use 40 or 60 burn-in steps, the training time is still competitive, compared with the variational vSLDA. For parallel-gSLDA+ using GraphLab, the training is consistently about 2 orders of magnitudes faster. Meanwhile, the classification performance is also comparable with that of gSLDA+, when the number of burn-in steps is larger than 40. In the previous experiments, we have set  $M = 40$  for both gSLDA+ and parallel-gSLDA+.

**Regularization constant  $c$ :** Fig. 5 shows the performance of gSLDA in the binary classification task with different  $c$  values. We can see that in a wide range, e.g., from 9 to 100, the performance is quite stable for all the three  $K$  values. But for the standard sLDA model, i.e.,  $c = 1$ , both the training accuracy and test accuracy are low, which indicates that sLDA doesn't fit the supervision data well. When  $c$  becomes larger, the training accuracy gets higher, but it doesn't seem to over-fit and the generalization performance is stable. In the above experiments, we set  $c = 25$ . For multi-class classification, we have similar observations and set  $c = 256$  in the previous experiments.

**Dirichlet prior  $\alpha$ :** Fig. 6 shows the performance of gSLDA on the binary task with different  $\alpha$  values. We report two cases with  $c = 1$  and  $c = 9$ . We can see that the performance is quite stable in a wide range of  $\alpha$  values, e.g., from 0.1

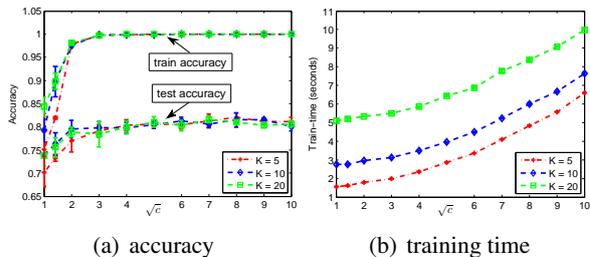


Figure 5: Performance of gSLDA for binary classification with different  $c$  values.

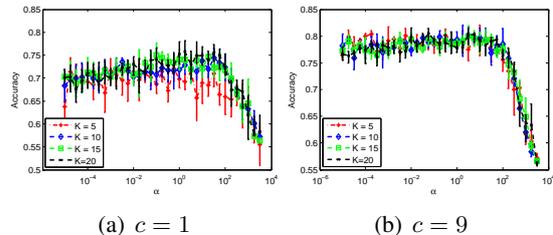


Figure 6: Accuracy of gSLDA for binary classification with different  $\alpha$  values in two settings with  $c = 1$  and  $c = 9$ .

to 10. We also noted that the change of  $\alpha$  does not affect the training time much.

## 5 Conclusions and Discussions

We present two improvements to Bayesian logistic supervised topic models, namely, a general formulation by introducing a regularization parameter to avoid model imbalance and a highly efficient Gibbs sampling algorithm without restricting assumptions on the posterior distributions by exploring the idea of data augmentation. The algorithm can also be parallelized. Empirical results for both binary and multi-class classification demonstrate significant improvements over the existing logistic supervised topic models. Our preliminary results with GraphLab have shown promise on parallelizing the Gibbs sampling algorithm.

For future work, we plan to carry out more careful investigations, e.g., using various distributed architectures (Ahmed et al., 2012; Newman et al., 2009; Smola and Narayanamurthy, 2010), to make the sampling algorithm highly scalable to deal with massive data corpora. Moreover, the data augmentation technique can be applied to deal with other types of response variables, such as count data with a negative-binomial likelihood (Polson et al., 2012).

## Acknowledgments

This work is supported by National Key Foundation R&D Projects (No.s 2013CB329403,

2012CB316301), Tsinghua Initiative Scientific Research Program No.20121088071, Tsinghua National Laboratory for Information Science and Technology, and the 221 Basic Research Plan for Young Faculties at Tsinghua University.

## References

- A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. Smola. 2012. Scalable inference in latent variable models. In *International Conference on Web Search and Data Mining (WSDM)*.
- D.M. Blei and J.D. McAuliffe. 2010. Supervised topic models. *arXiv:1003.0783v1*.
- D.M. Blei, A.Y. Ng, and M.I. Jordan. 2003. Latent Dirichlet allocation. *JMLR*, 3:993–1022.
- M. Chen, J. Ibrahim, and C. Yiannoutsos. 1999. Prior elicitation, variable selection and Bayesian computation for logistic regression models. *Journal of Royal Statistical Society, Ser. B*, (61):223–242.
- P. Germain, A. Lacasse, F. Laviolette, and M. Marchand. 2009. PAC-Bayesian learning of linear classifiers. In *International Conference on Machine Learning (ICML)*, pages 353–360.
- A. Globerson, T. Koo, X. Carreras, and M. Collins. 2007. Exponentiated gradient algorithms for log-linear structured prediction. In *ICML*, pages 305–312.
- J.E. Gonzalez, Y. Low, H. Gu, D. Bickson, and C. Guestrin. 2012. Powergraph: Distributed graph-parallel computation on natural graphs. In *the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.
- T.L. Griffiths and M. Steyvers. 2004. Finding scientific topics. *Proceedings of National Academy of Science (PNAS)*, pages 5228–5235.
- Y. Halpern, S. Horng, L. Nathanson, N. Shapiro, and D. Sontag. 2012. A comparison of dimensionality reduction techniques for unstructured clinical text. In *ICML 2012 Workshop on Clinical Data Analysis*.
- C. Holmes and L. Held. 2006. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, 1(1):145–168.
- Q. Jiang, J. Zhu, M. Sun, and E.P. Xing. 2012. Monte Carlo methods for maximum margin supervised topic models. In *Advances in Neural Information Processing Systems (NIPS)*.
- T. Joachims. 1999. *Making large-scale SVM learning practical*. MIT press.
- S. Lacoste-Jullien, F. Sha, and M.I. Jordan. 2009. DiscLDA: Discriminative learning for dimensionality reduction and classification. *Advances in Neural Information Processing Systems (NIPS)*, pages 897–904.
- Y. Lin. 2001. A note on margin-based loss functions in classification. *Technical Report No. 1044. University of Wisconsin*.
- D. McAllester. 2003. PAC-Bayesian stochastic model selection. *Machine Learning*, 51:5–21.
- M. Meyer and P. Laud. 2002. Predictive variable selection in generalized linear models. *Journal of American Statistical Association*, 97(459):859–871.
- D. Newman, A. Asuncion, P. Smyth, and M. Welling. 2009. Distributed algorithms for topic models. *Journal of Machine Learning Research (JMLR)*, (10):1801–1828.
- N.G. Polson, J.G. Scott, and J. Windle. 2012. Bayesian inference for logistic models using Polya-Gamma latent variables. *arXiv:1205.0310v1*.
- R. Rifkin and A. Klautau. 2004. In defense of one-vs-all classification. *Journal of Machine Learning Research (JMLR)*, (5):101–141.
- L. Rosasco, E. De Vito, A. Caponnetto, M. Piana, and A. Verri. 2004. Are loss functions all the same? *Neural Computation*, (16):1063–1076.
- A. Smola and S. Narayanamurthy. 2010. An architecture for parallel topic models. *Very Large Data Base (VLDB)*, 3(1-2):703–710.
- M.A. Tanner and W.-H. Wong. 1987. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association (JASA)*, 82(398):528–540.
- D. van Dyk and X. Meng. 2001. The art of data augmentation. *Journal of Computational and Graphical Statistics (JCGS)*, 10(1):1–50.
- C. Wang, D.M. Blei, and Li F.F. 2009. Simultaneous image classification and annotation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- J. Zhu, N. Chen, and E.P. Xing. 2011. Infinite latent SVM for classification and multi-task learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1620–1628.
- J. Zhu, A. Ahmed, and E.P. Xing. 2012. MedLDA: maximum margin supervised topic models. *Journal of Machine Learning Research (JMLR)*, (13):2237–2278.
- J. Zhu, N. Chen, H. Perkins, and B. Zhang. 2013a. Gibbs max-margin topic models with fast sampling algorithms. In *International Conference on Machine Learning (ICML)*.
- J. Zhu, N. Chen, and E.P. Xing. 2013b. Bayesian inference with posterior regularization and applications to infinite latent svms. *arXiv:1210.1766v2*.

# Fast and Robust Compressive Summarization with Dual Decomposition and Multi-Task Learning

Miguel B. Almeida\*<sup>†</sup>      André F. T. Martins\*<sup>†</sup>

\*Priberam Labs, Alameda D. Afonso Henriques, 41, 2<sup>o</sup>, 1000-123 Lisboa, Portugal

<sup>†</sup>Instituto de Telecomunicações, Instituto Superior Técnico, 1049-001 Lisboa, Portugal

{mba, atm}@priberam.pt

## Abstract

We present a dual decomposition framework for multi-document summarization, using a model that jointly extracts and compresses sentences. Compared with previous work based on integer linear programming, our approach does not require external solvers, is significantly faster, and is modular in the three qualities a summary should have: conciseness, informativeness, and grammaticality. In addition, we propose a multi-task learning framework to take advantage of existing data for extractive summarization and sentence compression. Experiments in the TAC-2008 dataset yield the highest published ROUGE scores to date, with runtimes that rival those of extractive summarizers.

## 1 Introduction

Automatic text summarization is a seminal problem in information retrieval and natural language processing (Luhn, 1958; Baxendale, 1958; Edmundson, 1969). Today, with the overwhelming amount of information available on the Web, the demand for fast, robust, and scalable summarization systems is stronger than ever.

Up to now, **extractive systems** have been the most popular in multi-document summarization. These systems produce a summary by extracting a representative set of sentences from the original documents (Kupiec et al., 1995; Carbonell and Goldstein, 1998; Radev et al., 2000; Gillick et al., 2008). This approach has obvious advantages: it reduces the search space by letting decisions be made for each sentence as a whole (avoiding fine-grained text generation), and it ensures a grammatical summary, assuming the original sentences are well-formed. The typical trade-offs in these mod-

els (maximizing relevance, and penalizing redundancy) lead to submodular optimization problems (Lin and Bilmes, 2010), which are NP-hard but approximable through greedy algorithms; learning is possible with standard structured prediction algorithms (Sipos et al., 2012; Lin and Bilmes, 2012). Probabilistic models have also been proposed to capture the problem structure, such as determinantal point processes (Gillenwater et al., 2012).

However, extractive systems are rather limited in the summaries they can produce. Long, partly relevant sentences tend not to appear in the summary, or to block the inclusion of other sentences. This has motivated research in **compressive summarization** (Lin, 2003; Zajic et al., 2006; Daumé, 2006), where summaries are formed by compressed sentences (Knight and Marcu, 2000), not necessarily extracts. While promising results have been achieved by models that simultaneously extract and compress (Martins and Smith, 2009; Woodsend and Lapata, 2010; Berg-Kirkpatrick et al., 2011), there are still obstacles that need to be surmounted for these systems to enjoy wide adoption. All approaches above are based on **integer linear programming** (ILP), suffering from slow runtimes, when compared to extractive systems. For example, Woodsend and Lapata (2012) report 55 seconds on average to produce a summary; Berg-Kirkpatrick et al. (2011) report substantially faster runtimes, but fewer compressions are allowed. Having a compressive summarizer which is both fast and expressive remains an open problem. A second inconvenience of ILP-based approaches is that they do not exploit the modularity of the problem, since the declarative specification required by ILP solvers discards important structural information. For example, such solvers are unable to take advantage of efficient dynamic programming routines for sentence compression (McDonald, 2006).

This paper makes progress in two fronts:

- We derive a **dual decomposition** framework for extractive and compressive summarization (§2–3). Not only is this framework orders of magnitude more efficient than the ILP-based approaches, it also allows the three well-known metrics of summaries—conciseness, informativeness, and grammaticality—to be treated separately in a modular fashion (see Figure 1). We also contribute with a novel **knapsack factor**, along with a linear-time algorithm for the corresponding dual decomposition subproblem.
- We propose **multi-task learning** (§4) as a principled way to train compressive summarizers, using auxiliary data for extractive summarization and sentence compression. To this end, we adapt the framework of Evgeniou and Pontil (2004) and Daumé (2007) to train structured predictors that share some of their parts.

Experiments on TAC data (§5) yield state-of-the-art results, with runtimes similar to that of extractive systems. To our best knowledge, this had never been achieved by compressive summarizers.

## 2 Extractive Summarization

In **extractive summarization**, we are given a set of sentences  $\mathcal{D} := \{s_1, \dots, s_N\}$  belonging to one or more documents, and the goal is to extract a subset  $\mathcal{S} \subseteq \mathcal{D}$  that conveys a good summary of  $\mathcal{D}$  and whose total number of words does not exceed a prespecified budget  $B$ .

We use an indicator vector  $\mathbf{y} := \langle y_n \rangle_{n=1}^N$  to represent an extractive summary, where  $y_n = 1$  if  $s_n \in \mathcal{S}$ , and  $y_n = 0$  otherwise. Let  $L_n$  be the number of words of the  $n$ th sentence. By designing a quality score function  $g : \{0, 1\}^N \rightarrow \mathbb{R}$ , this can be cast as a global optimization problem with a knapsack constraint:

$$\begin{aligned} & \text{maximize} && g(\mathbf{y}) \\ & \text{w.r.t.} && \mathbf{y} \in \{0, 1\}^N \\ & \text{s.t.} && \sum_{n=1}^N L_n y_n \leq B. \end{aligned} \quad (1)$$

Intuitively, a good summary is one which selects sentences that individually convey “relevant” information, while collectively having small “redundancy.” This trade-off was explicitly modeled in early works through the notion of **maximal marginal relevance** (Carbonell and Goldstein, 1998; McDonald, 2007). An alternative

are **coverage-based models** (§2.1; Filatova and Hatzivassiloglou, 2004; Yih et al., 2007; Gillick et al., 2008), which seek a set of sentences that covers as many diverse “concepts” as possible; redundancy is automatically penalized since redundant sentences cover fewer concepts. Both models can be framed under the framework of submodular optimization (Lin and Bilmes, 2010), leading to greedy algorithms that have approximation guarantees. However, extending these models to allow for sentence compression (as will be detailed in §3) breaks the diminishing returns property, making submodular optimization no longer applicable.

### 2.1 Coverage-Based Summarization

Coverage-based extractive summarization can be formalized as follows. Let  $\mathcal{C}(\mathcal{D}) := \{c_1, \dots, c_M\}$  be a set of relevant **concept types** which are present in the original documents  $\mathcal{D}$ .<sup>1</sup> Let  $\sigma_m$  be a relevance score assigned to the  $m$ th concept, and let the set  $\mathcal{J}_m \subseteq \{1, \dots, N\}$  contain the indices of the sentences in which this concept occurs. Then, the following quality score function is defined:

$$g(\mathbf{y}) = \sum_{m=1}^M \sigma_m u_m(\mathbf{y}), \quad (2)$$

where  $u_m(\mathbf{y}) := \bigvee_{n \in \mathcal{J}_m} y_n$  is a Boolean function that indicates whether the  $m$ th concept is present in the summary. Plugging this into Eq. 1, one obtains the following Boolean optimization problem:

$$\begin{aligned} & \text{maximize} && \sum_{m=1}^M \sigma_m u_m \\ & \text{w.r.t.} && \mathbf{y} \in \{0, 1\}^N, \mathbf{u} \in \{0, 1\}^M \\ & \text{s.t.} && u_m = \bigvee_{n \in \mathcal{J}_m} y_n, \forall m \in [M] \\ & && \sum_{n=1}^N L_n y_n \leq B, \end{aligned} \quad (3)$$

where we used the notation  $[M] := \{1, \dots, M\}$ . This can be converted into an ILP and addressed with off-the-shelf solvers (Gillick et al., 2008). A drawback of this approach is that solving an ILP exactly is NP-hard. Even though existing commercial solvers can solve most instances with a moderate speed, they still exhibit poor worst-case behaviour; this is exacerbated when there is the need to combine an extractive component with other modules, as in compressive summarization (§3).

<sup>1</sup>Previous work has modeled concepts as events (Filatova and Hatzivassiloglou, 2004), salient words (Lin and Bilmes, 2010), and word bigrams (Gillick et al., 2008). In the sequel, we assume concepts are word  $k$ -grams, but our model can handle other representations, such as phrases or predicate-argument structures.

## 2.2 A Dual Decomposition Formulation

We next describe how the problem in Eq. 3 can be addressed with **dual decomposition**, a class of optimization techniques that tackle the dual of combinatorial problems in a modular, extensible, and parallelizable manner (Komodakis et al., 2007; Rush et al., 2010). In particular, we employ **alternating directions dual decomposition** (AD<sup>3</sup>; Martins et al., 2011a, 2012) for solving a linear relaxation of Eq. 3. AD<sup>3</sup> resembles the subgradient-based algorithm of Rush et al. (2010), but it enjoys a faster convergence rate. Both algorithms split the original problem into several **components**, and then iterate between solving independent **local subproblems** at each component and adjusting multipliers to promote an agreement.<sup>2</sup> The difference between the two methods is that the AD<sup>3</sup> local subproblems, instead of requiring the computation of a locally optimal configuration, require solving a local *quadratic* problem. Martins et al. (2011b) provided linear-time solutions for several logic constraints, with applications to syntax and frame-semantic parsing (Das et al., 2012). We will see that AD<sup>3</sup> can also handle budget and knapsack constraints efficiently.

To tackle Eq. 3 with dual decomposition, we split the coverage-based summarizer into the following  $M + 1$  components (one per constraint):

1. For each of the  $M$  concepts in  $\mathcal{C}(\mathcal{D})$ , one component for imposing the logic constraint in Eq. 3. This corresponds to the OR-WITH-OUTPUT factor described by Martins et al. (2011b); the AD<sup>3</sup> subproblem for the  $m$ th factor can be solved in time  $O(|J_m|)$ .
2. Another component for the knapsack constraint. This corresponds to a (novel) KNAPSACK factor, whose AD<sup>3</sup> subproblem is solvable in time  $O(N)$ . The actual algorithm is described in the appendix (Algorithm 1).<sup>3</sup>

## 3 Compressive Summarization

We now turn to **compressive summarization**, which does not limit the summary sentences to be verbatim extracts from the original documents; in-

<sup>2</sup>For details about dual decomposition and Lagrangian relaxation, see the recent tutorial by Rush and Collins (2012).

<sup>3</sup>The AD<sup>3</sup> subproblem in this case corresponds to computing an Euclidean projection onto the knapsack polytope (Eq. 11). Others addressed the related, but much harder, integer quadratic knapsack problem (McDonald, 2007).

stead, it allows the extraction of *compressed* sentences where some words can be deleted.

Formally, let us express each sentence of  $\mathcal{D}$  as a sequence of word tokens,  $s_n := \langle t_{n,\ell} \rangle_{\ell=0}^{L_n}$ , where  $t_{n,0} \equiv \$$  is a dummy symbol. We represent a **compression** of  $s_n$  as an indicator vector  $z_n := \langle z_{n,\ell} \rangle_{\ell=0}^{L_n}$ , where  $z_{n,\ell} = 1$  if the  $\ell$ th word is included in the compression. By convention, the dummy symbol is included if and only if the remaining compression is non-empty. A compressive summary can then be represented by an indicator vector  $z$  which is the concatenation of  $N$  such vectors,  $z = \langle z_1, \dots, z_N \rangle$ ; each position in this indicator vector is indexed by a sentence  $n \in [N]$  and a word position  $\ell \in \{0\} \cup [L_n]$ .

Models for compressive summarization were proposed by Martins and Smith (2009) and Berg-Kirkpatrick et al. (2011) by combining extraction and compression scores. Here, we follow the latter work, by combining a coverage score function  $g$  with sentence-level compression score functions  $h_1, \dots, h_N$ . This yields the decoding problem:

$$\begin{aligned} & \text{maximize} && g(z) + \sum_{n=1}^N h_n(z_n) \\ & \text{w.r.t.} && z_n \in \{0, 1\}^{L_n}, \forall n \in [N] \\ & \text{s.t.} && \sum_{n=1}^N \sum_{\ell=1}^{L_n} z_{n,\ell} \leq B. \end{aligned} \quad (4)$$

### 3.1 Coverage Model

We use a coverage function similar to Eq. 2, but taking a compressive summary  $z$  as argument:

$$g(z) = \sum_{m=1}^M \sigma_m u_m(z), \quad (5)$$

where we redefine  $u_m$  as follows. First, we parametrize each occurrence of the  $m$ th concept (assumed to be a  $k$ -gram) as a triple  $\langle n, \ell_s, \ell_e \rangle$ , where  $n$  indexes a sentence,  $\ell_s$  indexes a start position within the sentence, and  $\ell_e$  indexes the end position. We denote by  $\mathcal{T}_m$  the set of triples representing all occurrences of the  $m$ th concept in the original text, and we associate an indicator variable  $z_{n,\ell_s:\ell_e}$  to each member of this set. We then define  $u_m(z)$  via the following logic constraints:

- A **concept type** is selected if *some* of its  $k$ -gram tokens are selected:

$$u_m(\mathbf{y}) := \bigvee_{\langle n, \ell_s, \ell_e \rangle \in \mathcal{T}_m} z_{n, \ell_s: \ell_e}. \quad (6)$$

- A  $k$ -gram **concept token** is selected if *all* its words are selected:

$$z_{n, \ell_s: \ell_e} := \bigwedge_{\ell=\ell_s}^{\ell_e} z_{n, \ell}. \quad (7)$$

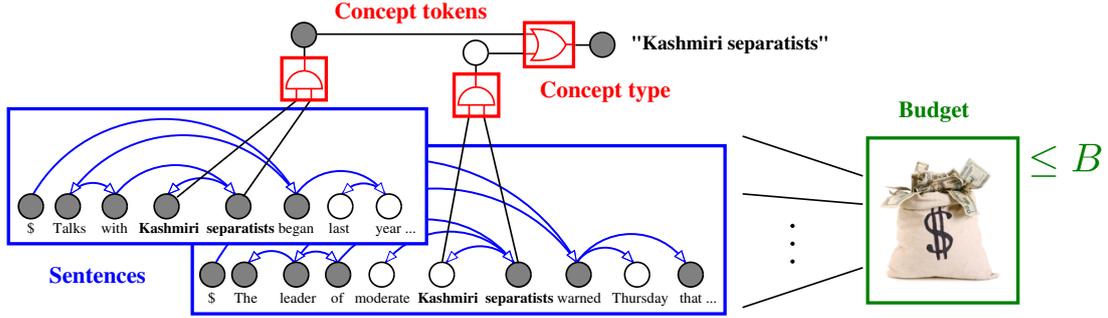


Figure 1: Components of our compressive summarizer. Factors depicted in blue belong to the compression model, and aim to enforce grammaticality. The logic factors in red form the coverage component. Finally, the budget factor, in green, is connected to the word nodes; it ensures that the summary fits the word limit. Shaded circles represent active variables while white circles represent inactive variables.

We set concept scores as  $\sigma_m := \mathbf{w} \cdot \Phi_{\text{cov}}(\mathcal{D}, c_m)$ , where  $\Phi_{\text{cov}}(\mathcal{D}, c_m)$  is a vector of features (described in §3.5) and  $\mathbf{w}$  the corresponding weights.

### 3.2 Compression Model

For the compression score function, we follow Martins and Smith (2009) and decompose it as a sum of local score functions  $\rho_{n,\ell}$  defined on dependency arcs:

$$h_n(\mathbf{z}_n) := \sum_{\ell=1}^{L_n} \rho_{n,\ell}(z_{n,\ell}, z_{n,\pi(\ell)}), \quad (8)$$

where  $\pi(\ell)$  denotes the index of the word which is the parent of the  $\ell$ th word in the dependency tree (by convention, the root of the tree is the dummy symbol). To model the event that an arc is “cut” by disconnecting a child from its head, we define **arc-deletion scores**  $\rho_{n,\ell}(0, 1) := \mathbf{w} \cdot \Phi_{\text{comp}}(s_n, \ell, \pi(\ell))$ , where  $\Phi_{\text{comp}}$  is a feature map, which is described in detail in §3.5. We set  $\rho_{n,\ell}(0, 0) = \rho_{n,\ell}(1, 1) = 0$ , and  $\rho_{n,\ell}(1, 0) = -\infty$ , to allow only the deletion of entire subtrees.

A crucial fact is that one can maximize Eq. 8 efficiently with dynamic programming (using the Viterbi algorithm for trees); the total cost is linear in  $L_n$ . We will exploit this fact in the dual decomposition framework described next.<sup>4</sup>

### 3.3 A Dual Decomposition Formulation

In previous work, the optimization problem in Eq. 4 was converted into an ILP and fed to an off-the-shelf solver (Martins and Smith, 2009; Berg-Kirkpatrick et al., 2011; Woodsend and Lapata, 2012). Here, we employ the AD<sup>3</sup> algorithm, in a

<sup>4</sup>The same framework can be readily adapted to other compression models that are efficiently decodable, such as the semi-Markov model of McDonald (2006), which would allow incorporating a language model for the compression.

similar manner as described in §2, but with an additional component for the sentence compressor, and slight modifications in the other components. We have the following  $N + M + \sum_{m=1}^M |\mathcal{T}_m| + 1$  components in total, illustrated in Figure 1:

1. For each of the  $N$  sentences, one component for the **compression model**. The AD<sup>3</sup> quadratic subproblem for this factor can be addressed by solving a *sequence of linear subproblems*, as described by Martins et al. (2012). Each of these subproblems corresponds to maximizing an objective function of the same form as Eq. 8; this can be done in  $O(L_n)$  time with dynamic programming, as discussed in §3.2.
2. For each of the  $M$  concept types in  $\mathcal{C}(\mathcal{D})$ , one OR-WITH-OUTPUT factor for the logic constraint in Eq. 6. This is analogous to the one described for the extractive case.
3. For each  $k$ -gram concept token in  $\mathcal{T}_m$ , one AND-WITH-OUTPUT factor that imposes the constraint in Eq. 7. This factor was described by Martins et al. (2011b) and its AD<sup>3</sup> subproblem can be solved in time linear in  $k$ .
4. Another component linked to all the words imposing that at most  $B$  words can be selected; this is done via a BUDGET factor, a particular case of KNAPSACK. The runtime of this AD<sup>3</sup> subproblem is linear in the number of words.

In addition, we found it useful to add a second BUDGET factor limiting the number of sentences that can be selected to a prescribed value  $K$ . We set  $K = 6$  in our experiments.

### 3.4 Rounding Strategy

Recall that the problem in Eq. 4 is NP-hard, and that AD<sup>3</sup> is solving a linear relaxation. While there are ways of wrapping AD<sup>3</sup> in an exact search algorithm (Das et al., 2012), such strategies work best when the solution of the relaxation has few fractional components, which is typical of parsing and translation problems (Rush et al., 2010; Chang and Collins, 2011), and attractive networks (Taskar et al., 2004). Unfortunately, this is not the case in summarization, where concepts “compete” with each other for inclusion in the summary, leading to frustrated cycles. We chose instead to adopt a fast and simple rounding procedure for obtaining a summary from a fractional solution.

The procedure works as follows. First, solve the LP relaxation using AD<sup>3</sup>, as described above. This yields a solution  $z^*$ , where each component lies in the unit interval  $[0, 1]$ . If these components are all integer, then we have a certificate that this is the optimal solution. Otherwise, we collect the  $K$  sentences with the highest values of  $z_{n,0}^*$  (“posteriors” on sentences), and seek the feasible summary which is the closest (in Euclidean distance) to  $z^*$ , while only containing those sentences. This can be computed exactly in time  $O(B \sum_{k=1}^K L_{n_k})$ , through dynamic programming.<sup>5</sup>

### 3.5 Features and Hard Constraints

As Berg-Kirkpatrick et al. (2011), we used stemmed word bigrams as concepts, to which we associate the following concept features ( $\Phi_{\text{cov}}$ ): indicators for document counts, features indicating if each of the words in the bigram is a stop-word, the earliest position in a document each concept occurs, as well as two and three-way conjunctions of these features.

For the compression model, we include the following arc-deletion features ( $\Phi_{\text{comp}}$ ):

- the dependency label of the arc being deleted, as well as its conjunction with the part-of-speech tag of the head, of the modifier, and of both;
- the dependency labels of the arc being deleted and of its parent arc;
- the modifier tag, if the modifier is a function word modifying a verb ;

<sup>5</sup>Briefly, if we link the roots of the  $K$  sentences to a super-root node, the problem above can be transformed into that of finding the best configuration in the resulting binary tree subject to a budget constraint. We omit details for space.

- a feature indicating whether the modifier or any of its descendants is a negation word;
- indicators of whether the modifier is a temporal word (e.g., *Friday*) or a preposition pointing to a temporal word (e.g., *on Friday*).

In addition, we included hard constraints to prevent the deletion of certain arcs, following previous work in sentence compression (Clarke and Lapata, 2008). We never delete arcs whose dependency label is SUB, OBJ, PMOD, SBAR, VC, or PRD (this makes sure we preserve subjects and objects of verbs, arcs departing from prepositions or complementizers, and that we do not break verb chains or predicative complements); arcs linking to a conjunction word or siblings of such arcs (to prevent inconsistencies in handling coordinative conjunctions); arcs linking verbs to other verbs, to adjectives (e.g., *make available*), to verb particles (e.g., *settle down*), to the word *that* (e.g., *said that*), or to the word *to* if it is a leaf (e.g., *allowed to come*); arcs pointing to negation words, cardinal numbers, or determiners; and arcs connecting two proper nouns or words within quotation marks.

## 4 Multi-Task Learning

We next turn to the problem of learning the model from training data. Prior work in compressive summarization has followed one of two strategies: Martins and Smith (2009) and Woodsend and Lapata (2012) learn the extraction and compression models separately, and then post-combine them, circumventing the lack of fully annotated data. Berg-Kirkpatrick et al. (2011) gathered a small dataset of manually compressed summaries, and trained with full supervision. While the latter approach is statistically more principled, it has the disadvantage of requiring fully annotated data, which is difficult to obtain in large quantities. On the other hand, there is plenty of data containing manually written abstracts (from the DUC and TAC conferences) and user-generated text (from Wikipedia) that may provide useful weak supervision.

With this in mind, we put together a **multi-task learning** framework for compressive summarization (which we name task #1). The goal is to take advantage of existing data for related tasks, such as extractive summarization (task #2), and sentence compression (task #3). The three tasks are instances of **structured predictors** (Bakır et

Tasks	Features	Decoder
Comp. summ. (#1)	$\Phi_{\text{cov}}, \Phi_{\text{comp}}$	AD <sup>3</sup> (solve Eq. 4)
Extr. summ. (#2)	$\Phi_{\text{cov}}$	AD <sup>3</sup> (solve Eq. 3)
Sent. comp. (#3)	$\Phi_{\text{comp}}$	dyn. prg. (max. Eq. 8)

Table 1: Features and decoders used for each task.

al., 2007), and for all of them we assume feature-based models that decompose over “parts”:

- For the compressive summarization task, the parts correspond to **concept features** (§3.1) and to **arc-deletion features** (§3.2).
- For the extractive summarization task, there are parts for **concept features** only.
- For the sentence compression task, the parts correspond to **arc-deletion features** only.

This is summarized in Table 1. Features for the three tasks are populated into feature vectors  $\Phi_1(x, y)$ ,  $\Phi_2(x, y)$ , and  $\Phi_3(x, y)$ , respectively, where  $\langle x, y \rangle$  denotes a task-specific input-output pair. We assume the feature vectors are all  $D$  dimensional, where we place zeros in entries corresponding to parts that are absent. Note that this setting is very general and applies to arbitrary structured prediction problems (not just summarization), the only assumption being that some parts are shared between different tasks.

Next, we associate weight vectors  $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in \mathbb{R}^D$  to each task, along with a “shared” vector  $\mathbf{w}$ . Each task makes predictions according to the rule:

$$\hat{y} := \arg \max_y (\mathbf{w} + \mathbf{v}_k) \cdot \Phi_k(x, y), \quad (9)$$

where  $k \in \{1, 2, 3\}$ . This setting is equivalent to the approach of Daumé (2007) for domain adaptation, which consists in splitting each feature into task-component features and a shared feature; but here we do not duplicate features explicitly. To learn the weights, we regularize the weight vectors separately, and assume that each task has its own loss function  $\mathcal{L}_k$ , so that the total loss  $\mathcal{L}$  is a weighted sum  $\mathcal{L}(\mathbf{w}, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3) := \sum_{k=1}^3 \sigma_k \mathcal{L}_k(\mathbf{w} + \mathbf{v}_k)$ . This yields the following objective function to be minimized:

$$F(\mathbf{w}, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3) = \frac{\lambda}{2} \|\mathbf{w}\|^2 + \sum_{k=1}^3 \frac{\lambda_k}{2} \|\mathbf{v}_k\|^2 + \frac{1}{N} \sum_{k=1}^3 \sigma_k \mathcal{L}_k(\mathbf{w} + \mathbf{v}_k), \quad (10)$$

where  $\lambda$  and the  $\lambda_k$ ’s are regularization constants, and  $N$  is the total number of training instances.<sup>6</sup> In our experiments (§5), we let the  $\mathcal{L}_k$ ’s be structured hinge losses (Taskar et al., 2003; Tsochantaridis et al., 2004), where the corresponding cost functions are concept recall (for task #2), precision of arc deletions (for task #3), and a combination thereof (for task #1).<sup>7</sup> These losses were normalized, and we set  $\sigma_k = N/N_k$ , where  $N_k$  is the number of training instances for the  $k$ th task. This ensures all tasks are weighted evenly. We used the same rationale to set  $\lambda = \lambda_1 = \lambda_2 = \lambda_3$ , choosing this value through cross-validation in the dev set.

We optimize Eq. 10 with stochastic subgradient descent. This leads to update rules of the form

$$\begin{aligned} \mathbf{w} &\leftarrow (1 - \eta_t \lambda) \mathbf{w} - \eta_t \sigma_k \tilde{\nabla} \mathcal{L}_k(\mathbf{w} + \mathbf{v}_k) \\ \mathbf{v}_j &\leftarrow (1 - \eta_t \lambda_j) \mathbf{v}_j - \eta_t \delta_{jk} \sigma_k \tilde{\nabla} \mathcal{L}_k(\mathbf{w} + \mathbf{v}_k), \end{aligned}$$

where  $\tilde{\nabla} \mathcal{L}_k$  are stochastic subgradients for the  $k$ th task, that take only a single instance into account, and  $\delta_{jk} = 1$  if and only if  $j = k$ . Stochastic subgradients can be computed via cost-augmented decoding (see footnote 7).

Interestingly, Eq. 10 subsumes previous approaches to train compressive summarizers. The limit  $\lambda \rightarrow \infty$  (keeping the  $\lambda_k$ ’s fixed) forces  $\mathbf{w} \rightarrow 0$ , decoupling all the tasks. In this limit, inference for task #1 (compressive summarization) is based solely on the model learned from that task’s data, recovering the approach of Berg-Kirkpatrick et al. (2011). In the other extreme, setting  $\sigma_1 = 0$  simply ignores task #1’s training data. As a result, the optimal  $\mathbf{v}_1$  will be a vector of zeros; since tasks #2 and #3 have no parts in common, the objective will decouple into a sum of two independent terms

<sup>6</sup>Note that, by substituting  $\mathbf{u}_k := \mathbf{w} + \mathbf{v}_k$  and solving for  $\mathbf{w}$ , the problem in Eq. 10 becomes that of minimizing the sum of the losses with a penalty for the (weighted) variance of the vectors  $\{\mathbf{0}, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ , regularizing the difference towards their average, as in Evgeniou and Pontil (2004). This is similar to the hierarchical joint learning approach of Finkel and Manning (2010), except that our goal is to learn a new task (compressive summarization) instead of combining tasks.

<sup>7</sup>Let  $\mathcal{Y}_k$  denote the output set for the  $k$ th task. Given a task-specific cost function  $\Delta_k : \mathcal{Y}_k \times \mathcal{Y}_k \rightarrow \mathbb{R}$ , and letting  $\langle x_t, y_t \rangle_{t=1}^T$  be the labeled dataset for this task, the structured hinge loss takes the form  $\mathcal{L}_k(\mathbf{u}_k) := \sum_t \max_{y' \in \mathcal{Y}_k} (\mathbf{u}_k \cdot (\Phi_k(x_t, y') - \Phi_k(x_t, y_t)) + \Delta_k(y', y_t))$ . The inner maximization over  $y'$  is called the *cost-augmented decoding problem*: it differs from Eq. 9 by the inclusion of the cost term  $\Delta_k(y', y_t)$ . Our costs decompose over the model’s factors, hence any decoder for Eq. 9 can be used for the maximization above: for tasks #1–#2, we solve a relaxation by running AD<sup>3</sup> without rounding, and for task #3 we use dynamic programming; see Table 1.

involving  $v_2$  and  $v_3$ , which is equivalent to training the two tasks separately and post-combining the models, as Martins and Smith (2009) did.

## 5 Experiments

### 5.1 Experimental setup

We evaluated our compressive summarizers on data from the Text Analysis Conference (TAC) evaluations. We use the same splits as previous work (Berg-Kirkpatrick et al., 2011; Woodsend and Lapata, 2012): the non-update portions of TAC-2009 for training and TAC-2008 for testing. In addition, we reserved TAC-2010 as a devset. The test partition contains 48 multi-document summarization problems; each provides 10 related news articles as input, and asks for a summary with up to 100 words, which is evaluated against four manually written abstracts. We ignored all the query information present in the TAC datasets.

**Single-Task Learning.** In the single-task experiments, we trained a compressive summarizer on the dataset disclosed by Berg-Kirkpatrick et al. (2011), which contains manual compressive summaries for the TAC-2009 data. We trained a structured SVM with stochastic subgradient descent; the cost-augmented inference problems are relaxed and solved with AD<sup>3</sup>, as described in §3.3.<sup>8</sup> We followed the procedure described in Berg-Kirkpatrick et al. (2011) to reduce the number of candidate sentences: scores were defined for each sentence (the sum of the scores of the concepts they cover), and the best-scored sentences were greedily selected up to a limit of 1,000 words. We then tagged and parsed the selected sentences with TurboParser.<sup>9</sup> Our choice of a dependency parser was motivated by our will for a fast system; in particular, TurboParser attains top accuracies at a rate of 1,200 words per second, keeping parsing times below 1 second for each summarization problem.

**Multi-Task Learning.** For the multi-task experiments, we also used the dataset of Berg-Kirkpatrick et al. (2011), but we augmented the training data with extractive summarization and sentence compression datasets, to help train the

compressive summarizer. For extractive summarization, we used the DUC 2003 and 2004 datasets (a total of 80 multi-document summarization problems). We generated oracle extracts by maximizing bigram recall with respect to the manual abstracts, as described in Berg-Kirkpatrick et al. (2011). For sentence compression, we adapted the Simple English Wikipedia dataset of Woodsend and Lapata (2011), containing aligned sentences for 15,000 articles from the English and Simple English Wikipedias. We kept only the 4,481 sentence pairs corresponding to deletion-based compressions.

### 5.2 Results

Table 2 shows the results. The top rows refer to three strong baselines: the ICSI-1 extractive coverage-based system of Gillick et al. (2008), which achieved the best ROUGE scores in the TAC-2008 evaluation; the compressive summarizer of Berg-Kirkpatrick et al. (2011), denoted BGK'11; and the multi-aspect compressive summarizer of Woodsend and Lapata (2012), denoted WL'12. All these systems require ILP solvers. The bottom rows show the results achieved by our implementation of a pure *extractive* system (similar to the learned extractive summarizer of Berg-Kirkpatrick et al., 2011); a system that *post-combines* extraction and compression components trained separately, as in Martins and Smith (2009); and our compressive summarizer trained as a *single task*, and in the *multi-task* setting.

The ROUGE and Pyramid scores show that the compressive summarizers (when properly trained) yield considerable benefits in content coverage over extractive systems, confirming the results of Berg-Kirkpatrick et al. (2011). Comparing the two bottom rows, we see a clear benefit by training in the multi-task setting, with a consistent gain in both coverage and linguistic quality. Our ROUGE-2 score (12.30%) is, to our knowledge, the highest reported on the TAC-2008 dataset, with little harm in grammaticality with respect to an extractive system that preserves the original sentences. Figure 2 shows an example summary.

### 5.3 Runtimes

We conducted another set of experiments to compare the runtime of our compressive summarizer based on AD<sup>3</sup> with the runtimes achieved by GLPK, the ILP solver used by Berg-Kirkpatrick et al. (2011). We varied the maximum number of it-

<sup>8</sup>We use the AD<sup>3</sup> implementation in <http://www.ark.cs.cmu.edu/AD3>, setting the maximum number of iterations to 200 at training time and 1000 at test time. We extended the code to handle the knapsack and budget factors; the modified code will be part of the next release (AD<sup>3</sup> 2.1).

<sup>9</sup><http://www.ark.cs.cmu.edu/TurboParser>

System	R-2	R-SU4	Pyr	LQ
ICSI-1	11.03	13.96	34.5 <sup>†</sup>	–
BGK’11	11.71	14.47	41.3 <sup>†</sup>	–
WL’12	11.37	14.47	–	–
Extractive	11.16	14.07	36.0	<b>4.6</b>
Post-comb.	11.07	13.85	38.4	4.1
Single-task	11.88	14.86	41.0	3.8
Multi-task	<b>12.30</b>	<b>15.18</b>	<b>42.6</b>	4.2

Table 2: Results for compressive summarization. Shown are the ROUGE-2 and ROUGE SU-4 recalls with the default options from the ROUGE toolkit (Lin, 2004); Pyramid scores (Nenkova and Passonneau, 2004); and linguistic quality scores, scored between 1 (very bad) to 5 (very good). For Pyramid, the evaluation was performed by two annotators, each evaluating half of the problems; scores marked with <sup>†</sup> were computed by different annotators and are not directly comparable. Linguistic quality was evaluated by two linguists; we show the average of the reported scores.

Solver	Runtime (sec.)	ROUGE-2
ILP Exact	10.394	12.40
LP-Relax.	2.265	12.38
AD <sup>3</sup> -5000	0.952	12.38
AD <sup>3</sup> -1000	0.406	12.30
AD <sup>3</sup> -200	0.159	12.15
Extractive (ILP)	0.265	11.16

Table 3: Runtimes of several decoders on a Intel Core i7 processor @2.8 GHz, with 8GB RAM. For each decoder, we show the average time taken to solve a summarization problem in TAC-2008. The reported runtimes of AD<sup>3</sup> and LP-Relax include the time taken to round the solution (§3.4), which is 0.029 seconds on average.

erations of AD<sup>3</sup> in {200, 1000, 5000}, and clocked the time spent by GLPK to solve the exact ILPs and their relaxations. Table 3 depicts the results.<sup>10</sup>

We see that our proposed configuration (AD<sup>3</sup>-1000) is orders of magnitude faster than the ILP solver, and 5 times faster than its relaxed variant, while keeping similar accuracy levels.<sup>11</sup> The gain when the number of iterations in AD<sup>3</sup> is increased to 5000 is small, given that the runtime is more

<sup>10</sup>Within dual decomposition algorithms, we verified experimentally that AD<sup>3</sup> is substantially faster than the subgradient algorithm, which is consistent with previous findings (Martins et al., 2011b).

<sup>11</sup>The runtimes obtained with the exact ILP solver seem slower than those reported by Berg-Kirkpatrick et al. (2011). (around 1.5 sec. on average, according to their Fig. 3). We conjecture that this difference is due to the restricted set of subtrees that can be deleted by Berg-Kirkpatrick et al. (2011), which greatly reduces their search space.

Japan dispatched four military ships to help Russia rescue seven crew members aboard a small submarine trapped on the seabed in the Far East. The Russian Pacific Fleet said the crew had 120 hours of oxygen reserves *on board when the submarine submerged at midday Thursday (2300 GMT Wednesday) off the Kamchatka peninsula, the stretch of Far Eastern Russia facing the Bering Sea.* The submarine, *used in rescue, research and intelligence-gathering missions,* became stuck at the bottom of the Bay of Berezovaya off Russia’s Far East coast when its propeller was caught *in a fishing net.* The Russian submarine had been tending an underwater antenna mounted to the sea floor *when it became snagged on a wire helping to stabilize a ventilation cable attached to the antenna.* Rescue crews lowered a British remote-controlled underwater vehicle to a Russian mini-submarine trapped *deep* under the Pacific Ocean, hoping to free the vessel and its seven trapped crewmen *before their air supply ran out.*

Figure 2: Example summary from our compressive system. Removed text is *grayed out*.

than doubled; accuracy starts to suffer, however, if the number of iterations is reduced too much. In practice, we observed that the final rounding procedure was crucial, as only 2 out of the 48 test problems had integral solutions (arguably because of the “repulsive” nature of the network, as hinted in §3.4). For comparison, we also report in the bottom row the average runtime of the learned extractive baseline. We can see that our system’s runtime is competitive with this baseline. To our knowledge, this is the first time a compressive summarizer achieves such a favorable accuracy/speed tradeoff.

## 6 Conclusions

We presented a multi-task learning framework for compressive summarization, leveraging data for related tasks in a principled manner. We decode with AD<sup>3</sup>, a fast and modular dual decomposition algorithm which is orders of magnitude faster than ILP-based approaches. Results show that the state of the art is improved in automatic and manual metrics, with speeds close to extractive systems.

Our approach is modular and easy to extend. For example, a different compression model could incorporate rewriting rules to enable compressions that go beyond word deletion, as in Cohn and Lapata (2008). Other aspects may be added as additional components in our dual decomposition framework, such as query information (Schilder and Kondadadi, 2008), discourse con-

straints (Clarke and Lapata, 2007), or lexical preferences (Woodsend and Lapata, 2012). Our multi-task approach may be used to jointly learn parameters for these aspects; the dual decomposition algorithm ensures that optimization remains tractable even with many components.

## A Projection Onto Knapsack

This section describes a linear-time algorithm (Algorithm 1) for solving the following problem:

$$\begin{aligned} & \text{minimize } \|\mathbf{z} - \mathbf{a}\|^2 \\ & \text{w.r.t. } z_n \in [0, 1], \forall n \in [N], \\ & \text{s.t. } \sum_{n=1}^N L_n z_n \leq B, \end{aligned} \quad (11)$$

where  $\mathbf{a} \in \mathbb{R}^N$  and  $L_n \geq 0, \forall n \in [N]$ . This includes as special cases the problems of projecting onto a budget constraint ( $L_n = 1, \forall n$ ) and onto the simplex (same, plus  $B = 1$ ).

Let  $\text{clip}(t) := \max\{0, \min\{1, t\}\}$ . Algorithm 1 starts by clipping  $\mathbf{a}$  to the unit interval; if that yields a  $\mathbf{z}$  satisfying  $\sum_{n=1}^N L_n z_n \leq B$ , we are done. Otherwise, the solution of Eq. 11 must satisfy  $\sum_{n=1}^N L_n z_n = B$ . It can be shown from the KKT conditions that the solution is of the form  $z_n^* := \text{clip}(a_n + \tau^* L_n)$  for a constant  $\tau^*$  lying in a particular interval of split-points (line 11). To seek this constant, we use an algorithm due to Pardalos and Koor (1990) which iteratively shrinks this interval. The algorithm requires computing medians as a subroutine, which can be done in linear time (Blum et al., 1973). The overall complexity is  $O(N)$  (Pardalos and Koor, 1990).

## Acknowledgments

We thank all reviewers for their insightful comments; Trevor Cohn for helpful discussions about multi-task learning; Taylor Berg-Kirkpatrick for answering questions about their summarizer and for providing code; and Helena Figueira and Pedro Mendes for helping with manual evaluation. This work was partially supported by the EU/FEDER programme, QREN/POR Lisboa (Portugal), under the Discooperio project (contract 2011/18501), and by a FCT grant PTDC/EEI-SII/2312/2012.

## References

G. Bakır, T. Hofmann, B. Schölkopf, A. Smola, B. Taskar, and S. Vishwanathan. 2007. *Predicting Structured Data*. The MIT Press.

---

## Algorithm 1 Projection Onto Knapsack.

---

```

1: input:  $\mathbf{a} := \langle a_n \rangle_{n=1}^N$ , costs  $\langle L_n \rangle_{n=1}^N$ , maximum cost  $B$ 
2:
3: {Try to clip into unit interval:}
4: Set  $z_n \leftarrow \text{clip}(a_n)$  for  $n \in [N]$ 
5: if  $\sum_{n=1}^N L_n z_n \leq B$  then
6:   Return  $\mathbf{z}$  and stop.
7: end if
8:
9: {Run Pardalos and Koor (1990)'s algorithm:}
10: Initialize working set  $\mathcal{W} \leftarrow \{1, \dots, K\}$ 
11: Initialize set of split points:
     $\mathcal{P} \leftarrow \{-a_n/L_n, (1 - a_n)/L_n\}_{n=1}^N \cup \{\pm\infty\}$ 
12: Initialize  $\tau_L \leftarrow -\infty, \tau_R \leftarrow \infty, s_{\text{tight}} \leftarrow 0, \xi \leftarrow 0$ .
13: while  $\mathcal{W} \neq \emptyset$  do
14:   Compute  $\tau \leftarrow \text{Median}(\mathcal{P})$ 
15:   Set  $s \leftarrow s_{\text{tight}} + \xi\tau + \sum_{n \in \mathcal{W}} L_n \text{clip}(a_n + \tau L_n)$ 
16:   If  $s \leq B$ , set  $\tau_L \leftarrow \tau$ ; if  $s \geq B$ , set  $\tau_R \leftarrow \tau$ 
17:   Reduce set of split points:  $\mathcal{P} \leftarrow \mathcal{P} \cap [\tau_L, \tau_R]$ 
18:   Define the sets:
     $\mathcal{W}_L := \{n \in \mathcal{W} \mid (1 - a_n)/L_n < \tau_L\}$ 
     $\mathcal{W}_R := \{n \in \mathcal{W} \mid -a_n/L_n > \tau_R\}$ 
     $\mathcal{W}_M := \left\{ n \in \mathcal{W} \mid -\frac{a_n}{L_n} \leq \tau_L \wedge \frac{1 - a_n}{L_n} \geq \tau_R \right\}$ 
19:   Update working set:  $\mathcal{W} \leftarrow \mathcal{W} \setminus (\mathcal{W}_L \cup \mathcal{W}_R \cup \mathcal{W}_M)$ 
20:   Update tight-sum:
     $s_{\text{tight}} \leftarrow s_{\text{tight}} + \sum_{n \in \mathcal{W}_L} L_n(1 - a_n) - \sum_{n \in \mathcal{W}_R} L_n a_n$ 
21:   Update slack-sum:  $\xi \leftarrow \xi + \sum_{n \in \mathcal{W}_M} L_n^2$ 
22: end while
23: Define  $\tau^* \leftarrow (B - \sum_{i=1}^N L_i a_i - s_{\text{tight}})/\xi$ 
24: Set  $z_n \leftarrow \text{clip}(a_n + \tau^* L_n), \forall n \in [N]$ 
25: output:  $\mathbf{z} := \langle z_n \rangle_{n=1}^N$ .

```

---

P. B. Baxendale. 1958. Machine-made index for technical literature—an experiment. *IBM Journal of Research Development*, 2(4):354–361.

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proc. of Annual Meeting of the Association for Computational Linguistics*.

Manuel Blum, Robert W Floyd, Vaughan Pratt, Ronald L Rivest, and Robert E Tarjan. 1973. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461.

J. Carbonell and J. Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*.

Y.-W. Chang and M. Collins. 2011. Exact decoding of phrase-based translation models through lagrangian relaxation. In *Proc. of Empirical Methods for Natural Language Processing*.

James Clarke and Mirella Lapata. 2007. Modelling compression with discourse constraints. In *Proc. of Empirical Methods in Natural Language Processing*.

J. Clarke and M. Lapata. 2008. Global Inference for Sentence Compression An Integer Linear Program-

- ming Approach. *Journal of Artificial Intelligence Research*, 31:399–429.
- T. Cohn and M. Lapata. 2008. Sentence compression beyond word deletion. In *Proc. COLING*.
- D. Das, A. F. T. Martins, and N. A. Smith. 2012. An Exact Dual Decomposition Algorithm for Shallow Semantic Parsing with Constraints. In *Proc. of First Joint Conference on Lexical and Computational Semantics (\*SEM)*.
- H. Daumé. 2006. *Practical Structured Learning Techniques for Natural Language Processing*. Ph.D. thesis, University of Southern California.
- H. Daumé. 2007. Frustratingly easy domain adaptation. In *Proc. of Annual Meeting of the Association for Computational Linguistics*.
- H. P. Edmundson. 1969. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285.
- T. Evgeniou and M. Pontil. 2004. Regularized multi-task learning. In *Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117. ACM.
- Elena Filatova and Vasileios Hatzivassiloglou. 2004. A formal model for information selection in multi-sentence text extraction. In *Proc. of International Conference on Computational Linguistics*.
- J.R. Finkel and C.D. Manning. 2010. Hierarchical joint learning: Improving joint parsing and named entity recognition with non-jointly labeled data. In *Proc. of Annual Meeting of the Association for Computational Linguistics*.
- J. Gillenwater, A. Kulesza, and B. Taskar. 2012. Discovering diverse and salient threads in document collections. In *Proc. of Empirical Methods in Natural Language Processing*.
- Dan Gillick, Benoit Favre, and Dilek Hakkani-Tur. 2008. The icsi summarization system at tac 2008. In *Proc. of Text Understanding Conference*.
- K. Knight and D. Marcu. 2000. Statistics-based summarization—step one: Sentence compression. In *AAAI/IAAI*.
- N. Komodakis, N. Paragios, and G. Tziritas. 2007. MRF optimization via dual decomposition: Message-passing revisited. In *Proc. of International Conference on Computer Vision*.
- J. Kupiec, J. Pedersen, and F. Chen. 1995. A trainable document summarizer. In *SIGIR*.
- H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Proc. of Annual Meeting of the North American chapter of the Association for Computational Linguistics*.
- H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Proc. of Uncertainty in Artificial Intelligence*.
- C.-Y. Lin. 2003. Improving summarization performance by sentence compression—a pilot study. In *the Int. Workshop on Inf. Ret. with Asian Languages*.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July.
- H. P. Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research Development*, 2(2):159–165.
- A. F. T. Martins and N. A. Smith. 2009. Summarization with a Joint Model for Sentence Extraction and Compression. In *North American Chapter of the Association for Computational Linguistics: Workshop on Integer Linear Programming for NLP*.
- A. F. T. Martins, M. A. T. Figueiredo, P. M. Q. Aguiar, N. A. Smith, and E. P. Xing. 2011a. An Augmented Lagrangian Approach to Constrained MAP Inference. In *Proc. of International Conference on Machine Learning*.
- A. F. T. Martins, N. A. Smith, P. M. Q. Aguiar, and M. A. T. Figueiredo. 2011b. Dual Decomposition with Many Overlapping Components. In *Proc. of Empirical Methods for Natural Language Processing*.
- Andre F. T. Martins, Mario A. T. Figueiredo, Pedro M. Q. Aguiar, Noah A. Smith, and Eric P. Xing. 2012. Alternating Directions Dual Decomposition. *Arxiv preprint arXiv:1212.6550*.
- R. McDonald. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proc. of Annual Meeting of the European Chapter of the Association for Computational Linguistics*.
- R. McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *ECIR*.
- A. Nenkova and R. Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of NAACL*, pages 145–152.
- Panos M. Pardalos and Naina Kooor. 1990. An algorithm for a singly constrained class of quadratic programs subject to upper and lower bounds. *Mathematical Programming*, 46(1):321–328.
- D. R. Radev, H. Jing, and M. Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *the NAACL-ANLP Workshop on Automatic Summarization*.

- A.M. Rush and M. Collins. 2012. A Tutorial on Dual Decomposition and Lagrangian Relaxation for Inference in Natural Language Processing. *Journal of Artificial Intelligence Research*, 45:305–362.
- A. Rush, D. Sontag, M. Collins, and T. Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proc. of Empirical Methods for Natural Language Processing*.
- Frank Schilder and Ravikumar Kondadadi. 2008. Fastsum: Fast and accurate query-based multi-document summarization. In *Proc. of Annual Meeting of the Association for Computational Linguistics*.
- R. Sipos, P. Shivaswamy, and T. Joachims. 2012. Large-margin learning of submodular summarization models.
- B. Taskar, C. Guestrin, and D. Koller. 2003. Max-margin Markov networks. In *Proc. of Neural Information Processing Systems*.
- B. Taskar, V. Chatalbashev, and D. Koller. 2004. Learning associative Markov networks. In *Proc. of International Conference of Machine Learning*.
- I. Tsochantaris, T. Hofmann, T. Joachims, and Y. Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proc. of International Conference of Machine Learning*.
- K. Woodsend and M. Lapata. 2010. Automatic generation of story highlights. In *Proc. of Annual Meeting of the Association for Computational Linguistics*, pages 565–574.
- Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proc. of Empirical Methods in Natural Language Processing*.
- Kristian Woodsend and Mirella Lapata. 2012. Multiple aspect summarization using integer linear programming. In *Proc. of Empirical Methods in Natural Language Processing*.
- Wen-tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. 2007. Multi-document summarization by maximizing informative content-words. In *Proc. of International Joint Conference on Artificial Intelligence*.
- D. Zajic, B. Dorr, J. Lin, and R. Schwartz. 2006. Sentence compression as a component of a multi-document summarization system. In *the ACL DUC Workshop*.

# Unsupervised Transcription of Historical Documents

Taylor Berg-Kirkpatrick Greg Durrett Dan Klein

Computer Science Division

University of California at Berkeley

{tberg, gdurrett, klein}@cs.berkeley.edu

## Abstract

We present a generative probabilistic model, inspired by historical printing processes, for transcribing images of documents from the printing press era. By jointly modeling the text of the document and the noisy (but regular) process of rendering glyphs, our unsupervised system is able to decipher font structure and more accurately transcribe images into text. Overall, our system substantially outperforms state-of-the-art solutions for this task, achieving a 31% relative reduction in word error rate over the leading commercial system for historical transcription, and a 47% relative reduction over Tesseract, Google’s open source OCR system.

## 1 Introduction

Standard techniques for transcribing modern documents do not work well on historical ones. For example, even state-of-the-art OCR systems produce word error rates of over 50% on the documents shown in Figure 1. Unsurprisingly, such error rates are too high for many research projects (Arlitsch and Herbert, 2004; Shoemaker, 2005; Holley, 2010). We present a new, generative model specialized to transcribing printing-press era documents. Our model is inspired by the underlying printing processes and is designed to capture the primary sources of variation and noise.

One key challenge is that the fonts used in historical documents are not standard (Shoemaker, 2005). For example, consider Figure 1a. The fonts are not irregular like handwriting – each occurrence of a given character type, e.g. *a*, will use the same underlying glyph. However, the exact glyphs are unknown. Some differences between fonts are minor, reflecting small variations in font design. Others are more severe, like the presence of the archaic long *s* character before 1804. To address the general problem of unknown fonts, our model

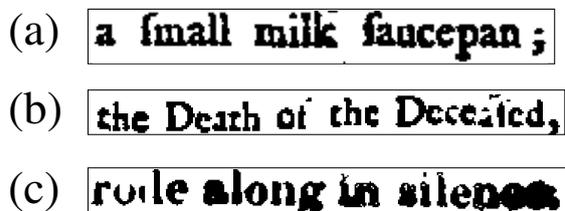


Figure 1: Portions of historical documents with (a) unknown font, (b) uneven baseline, and (c) over-inking.

learns the font in an unsupervised fashion. Font shape and character segmentation are tightly coupled, and so they are modeled jointly.

A second challenge with historical data is that the early typesetting process was noisy. Hand-carved blocks were somewhat uneven and often failed to sit evenly on the mechanical baseline. Figure 1b shows an example of the text’s baseline moving up and down, with varying gaps between characters. To deal with these phenomena, our model incorporates random variables that specifically describe variations in vertical offset and horizontal spacing.

A third challenge is that the actual inking was also noisy. For example, in Figure 1c some characters are thick from over-inking while others are obscured by ink bleeds. To be robust to such rendering irregularities, our model captures both inking levels and pixel-level noise. Because the model is generative, we can also treat areas that are obscured by larger ink blotches as unobserved, and let the model predict the obscured text based on visual and linguistic context.

Our system, which we call *Ocular*, operates by fitting the model to each document in an unsupervised fashion. The system outperforms state-of-the-art baselines, giving a 47% relative error reduction over Google’s open source Tesseract system, and giving a 31% relative error reduction over ABBYY’s commercial FineReader system, which has been used in large-scale historical transcription projects (Holley, 2010).

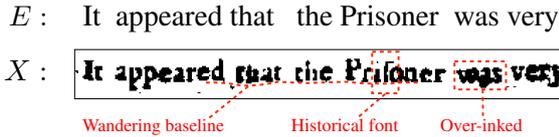


Figure 2: An example image from a historical document ( $X$ ) and its transcription ( $E$ ).

## 2 Related Work

Relatively little prior work has built models specifically for transcribing historical documents. Some of the challenges involved have been addressed (Ho and Nagy, 2000; Huang et al., 2006; Kae and Learned-Miller, 2009), but not in a way targeted to documents from the printing press era. For example, some approaches have learned fonts in an unsupervised fashion but require pre-segmentation of the image into character or word regions (Ho and Nagy, 2000; Huang et al., 2006), which is not feasible for noisy historical documents. Kae and Learned-Miller (2009) jointly learn the font and image segmentation but do not outperform modern baselines.

Work that has directly addressed historical documents has done so using a pipelined approach, and without fully integrating a strong language model (Vamvakas et al., 2008; Kluzner et al., 2009; Kae et al., 2010; Kluzner et al., 2011). The most comparable work is that of Kopec and Lomelin (1996) and Kopec et al. (2001). They integrated typesetting models with language models, but did not model noise. In the NLP community, generative models have been developed specifically for correcting outputs of OCR systems (Kolak et al., 2003), but these do not deal directly with images.

A closely related area of work is automatic decipherment (Ravi and Knight, 2008; Snyder et al., 2010; Ravi and Knight, 2011; Berg-Kirkpatrick and Klein, 2011). The fundamental problem is similar to our own: we are presented with a sequence of symbols, and we need to learn a correspondence between symbols and letters. Our approach is also similar in that we use a strong language model (in conjunction with the constraint that the correspondence be regular) to learn the correct mapping. However, the symbols are not noisy in decipherment problems and in our problem we face a grid of pixels for which the segmentation into symbols is unknown. In contrast, decipherment typically deals only with discrete symbols.

## 3 Model

Most historical documents have unknown fonts, noisy typesetting layouts, and inconsistent ink levels, usually simultaneously. For example, the portion of the document shown in Figure 2 has all three of these problems. Our model must handle them jointly.

We take a generative modeling approach inspired by the overall structure of the historical printing process. Our model generates images of documents line by line; we present the generative process for the image of a single line. Our primary random variables are  $E$  (the text) and  $X$  (the pixels in an image of the line). Additionally, we have a random variable  $T$  that specifies the layout of the bounding boxes of the glyphs in the image, and a random variable  $R$  that specifies aspects of the inking and rendering process. The joint distribution is:

$$\begin{aligned}
 P(E, T, R, X) = & \\
 & P(E) \quad \text{[Language model]} \\
 & \cdot P(T|E) \quad \text{[Typesetting model]} \\
 & \cdot P(R) \quad \text{[Inking model]} \\
 & \cdot P(X|E, T, R) \quad \text{[Noise model]}
 \end{aligned}$$

We let capital letters denote vectors of concatenated random variables, and we denote the individual random variables with lower-case letters. For example,  $E$  represents the entire sequence of text, while  $e_i$  represents  $i$ th character in the sequence.

### 3.1 Language Model $P(E)$

Our language model,  $P(E)$ , is a Kneser-Ney smoothed character  $n$ -gram model (Kneser and Ney, 1995). We generate printed lines of text (rather than sentences) independently, without generating an explicit stop character. This means that, formally, the model must separately generate the character length of each line. We choose not to bias the model towards longer or shorter character sequences and let the line length  $m$  be drawn uniformly at random from the positive integers less than some large constant  $M$ .<sup>1</sup> When  $i < 1$ , let  $e_i$  denote a line-initial null character. We can now write:

$$P(E) = P(m) \cdot \prod_{i=1}^m P(e_i | e_{i-1}, \dots, e_{i-n})$$

<sup>1</sup>In particular, we do not use the kind of “word bonus” common to statistical machine translation models.

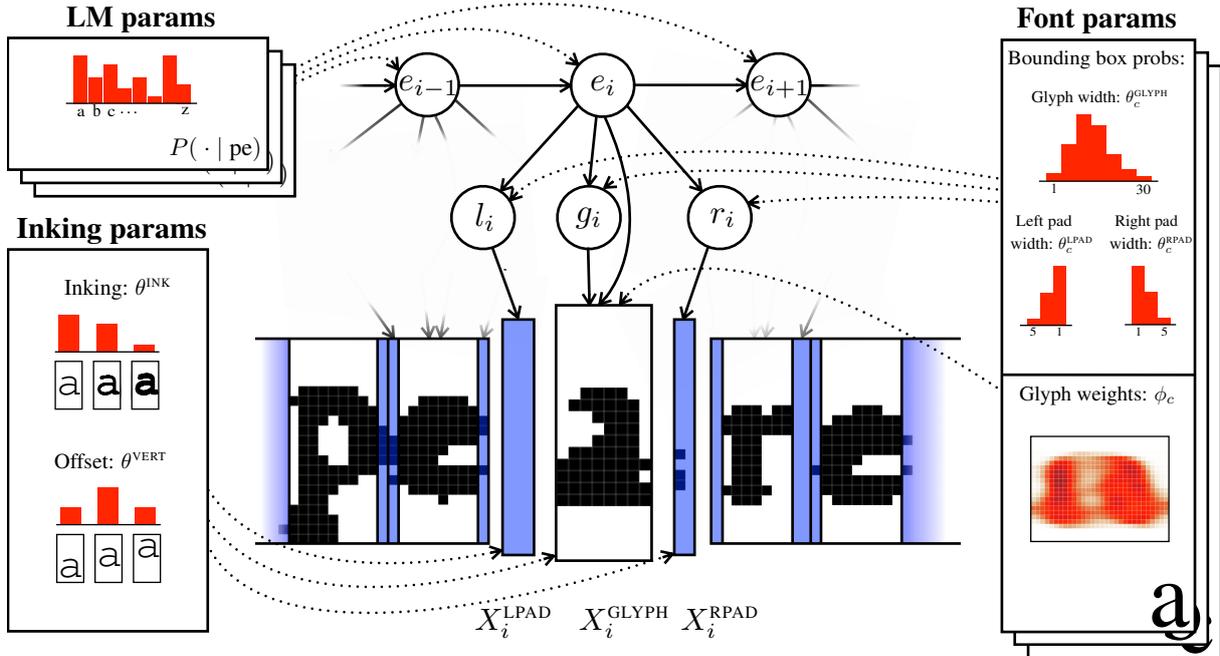


Figure 3: Character tokens  $e_i$  are generated by the language model. For each token index  $i$ , a glyph bounding box width  $g_i$ , left padding width  $l_i$ , and a right padding width  $r_i$ , are generated. Finally, the pixels in each glyph bounding box  $X_i^{\text{GLYPH}}$  are generated conditioned on the corresponding character, while the pixels in left and right padding bounding boxes,  $X_i^{\text{LPAD}}$  and  $X_i^{\text{RPAD}}$ , are generated from a background distribution.

### 3.2 Typesetting Model $P(T|E)$

Generally speaking, the process of typesetting produces a line of text by first tiling bounding boxes of various widths and then filling in the boxes with glyphs. Our generative model, which is depicted in Figure 3, reflects this process. As a first step, our model generates the dimensions of character bounding boxes; for each character token index  $i$  we generate three bounding box widths: a glyph box width  $g_i$ , a left padding box width  $l_i$ , and a right padding box width  $r_i$ , as shown in Figure 3. We let the pixel height of all lines be fixed to  $h$ . Let  $T_i = (l_i, g_i, r_i)$  so that  $T_i$  specifies the dimensions of the character box for token index  $i$ ;  $T$  is then the concatenation of all  $T_i$ , denoting the full layout.

Because the width of a glyph depends on its shape, and because of effects resulting from kerning and the use of ligatures, the components of each  $T_i$  are drawn conditioned on the character token  $e_i$ . This means that, as part of our parameterization of the font, for each character type  $c$  we have vectors of multinomial parameters  $\theta_c^{\text{LPAD}}$ ,  $\theta_c^{\text{GLYPH}}$ , and  $\theta_c^{\text{RPAD}}$  governing the distribution of the dimensions of character boxes of type  $c$ . These parameters are depicted on the right-hand side of

Figure 3. We can now express the typesetting layout portion of the model as:

$$\begin{aligned}
 P(T|E) &= \prod_{i=1}^m P(T_i|e_i) \\
 &= \prod_{i=1}^m [P(l_i; \theta_{e_i}^{\text{LPAD}}) \cdot P(g_i; \theta_{e_i}^{\text{GLYPH}}) \cdot P(r_i; \theta_{e_i}^{\text{RPAD}})]
 \end{aligned}$$

Each character type  $c$  in our font has another set of parameters, a matrix  $\phi_c$ . These are weights that specify the *shape* of the character type's glyph, and are depicted in Figure 3 as part of the font parameters.  $\phi_c$  will come into play when we begin generating pixels in Section 3.3.

#### 3.2.1 Inking Model $P(R)$

Before we start filling the character boxes with pixels, we need to specify some properties of the inking and rendering process, including the amount of ink used and vertical variation along the text baseline. Our model does this by generating, for each character token index  $i$ , a discrete value  $d_i$  that specifies the overall inking level in the character's bounding box, and a discrete value  $v_i$  that specifies the glyph's vertical offset. These variations in the inking and typesetting process are mostly independent of character type. Thus, in

our model, their distributions are not character-specific. There is one global set of multinomial parameters governing inking level ( $\theta^{\text{INK}}$ ), and another governing offset ( $\theta^{\text{VERT}}$ ); both are depicted on the left-hand side of Figure 3. Let  $R_i = (d_i, v_i)$  and let  $R$  be the concatenation of all  $R_i$  so that we can express the inking model as:

$$\begin{aligned} P(R) &= \prod_{i=1}^m P(R_i) \\ &= \prod_{i=1}^m [P(d_i; \theta^{\text{INK}}) \cdot P(v_i; \theta^{\text{VERT}})] \end{aligned}$$

The  $d_i$  and  $v_i$  variables are suppressed in Figure 3 to reduce clutter but are expressed in Figure 4, which depicts the process of rendering a glyph box.

### 3.3 Noise Model $P(X|E, T, R)$

Now that we have generated a typesetting layout  $T$  and an inking context  $R$ , we have to actually generate each of the pixels in each of the character boxes, left padding boxes, and right padding boxes; the matrices that these groups of pixels comprise are denoted  $X_i^{\text{GLYPH}}$ ,  $X_i^{\text{LPAD}}$ , and  $X_i^{\text{RPAD}}$ , respectively, and are depicted at the bottom of Figure 3.

We assume that pixels are binary valued and sample their values independently from Bernoulli distributions.<sup>2</sup> The probability of black (the Bernoulli parameter) depends on the type of pixel generated. All the pixels in a padding box have the same probability of black that depends only on the inking level of the box,  $d_i$ . Since we have already generated this value and the widths  $l_i$  and  $r_i$  of each padding box, we have enough information to generate left and right padding pixel matrices  $X_i^{\text{LPAD}}$  and  $X_i^{\text{RPAD}}$ .

The Bernoulli parameter of a pixel inside a glyph bounding box depends on the pixel’s location inside the box (as well as on  $d_i$  and  $v_i$ , but for simplicity of exposition, we temporarily suppress this dependence) and on the model parameters governing glyph shape (for each character type  $c$ , the parameter matrix  $\phi_c$  specifies the shape of the character’s glyph.) The process by which glyph pixels are generated is depicted in Figure 4.

The dependence of glyph pixels on location complicates generation of the glyph pixel matrix  $X_i^{\text{GLYPH}}$  since the corresponding parameter matrix

<sup>2</sup>We could generate real-valued pixels with a different choice of noise distribution.

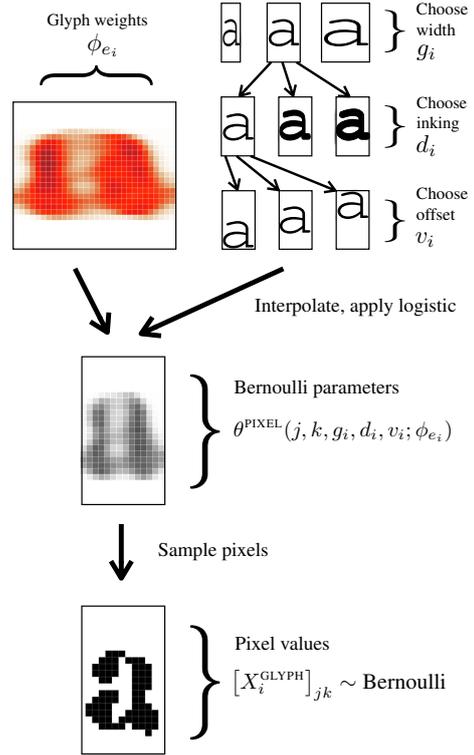


Figure 4: We generate the pixels for the character token  $e_i$  by first sampling a glyph width  $g_i$ , an inking level  $d_i$ , and a vertical offset  $v_i$ . Then we interpolate the glyph weights  $\phi_{e_i}$  and apply the logistic function to produce a matrix of Bernoulli parameters of width  $g_i$ , inking  $d_i$ , and offset  $v_i$ .  $\theta^{\text{PIXEL}}(j, k, g_i, d_i, v_i; \phi_{e_i})$  is the Bernoulli parameter at row  $j$  and column  $k$ . Finally, we sample from each Bernoulli distribution to generate a matrix of pixel values,  $X_i^{\text{GLYPH}}$ .

$\phi_{e_i}$  has some type-level width  $w$  which may differ from the current token-level width  $g_i$ . Introducing distinct parameters for each possible width would yield a model that can learn completely different glyph shapes for slightly different widths of the same character. We, instead, need a parameterization that ties the shapes for different widths together, and at the same time allows mobility in the parameter space during learning.

Our solution is to horizontally interpolate the weights of the shape parameter matrix  $\phi_{e_i}$  down to a smaller set of columns matching the token-level choice of glyph width  $g_i$ . Thus, the type-level matrix  $\phi_{e_i}$  specifies the canonical shape of the glyph for character  $e_i$  when it takes its maximum width  $w$ . After interpolating, we apply the logistic function to produce the individual Bernoulli parameters. If we let  $[X_i^{\text{GLYPH}}]_{jk}$  denote the value of the pixel at the  $j$ th row and  $k$ th column of the glyph pixel matrix  $X_i^{\text{GLYPH}}$  for token  $i$ , and let  $\theta^{\text{PIXEL}}(j, k, g_i; \phi_{e_i})$  denote the token-level

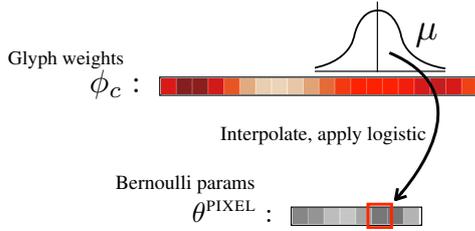


Figure 5: In order to produce Bernoulli parameter matrices  $\theta^{\text{PIXEL}}$  of variable width, we interpolate over columns of  $\phi_c$  with vectors  $\mu$ , and apply the logistic function to each result.

Bernoulli parameter for this pixel, we can write:

$$[X_i^{\text{GLYPH}}]_{jk} \sim \text{Bernoulli}(\theta^{\text{PIXEL}}(j, k, g_i; \phi_{e_i}))$$

The interpolation process for a single row is depicted in Figure 5. We define a constant interpolation vector  $\mu(g_i, k)$  that is specific to the glyph box width  $g_i$  and glyph box column  $k$ . Each  $\mu(g_i, k)$  is shaped according to a Gaussian centered at the relative column position in  $\phi_{e_i}$ . The glyph pixel Bernoulli parameters are defined as follows:

$$\theta^{\text{PIXEL}}(j, k, g_i; \phi_{e_i}) = \text{logistic} \left( \sum_{k'=1}^w [\mu(g_i, k)_{k'} \cdot [\phi_{e_i}]_{jk'}] \right)$$

The fact that the parameterization is log-linear will ensure that, during the unsupervised learning process, updating the shape parameters  $\phi_c$  is simple and feasible.

By varying the magnitude of  $\mu$  we can change the level of smoothing in the logistic model and cause it to permit areas that are over-inked. This is the effect that  $d_i$  controls. By offsetting the rows of  $\phi_c$  that we interpolate weights from, we change the vertical offset of the glyph, which is controlled by  $v_i$ . The full pixel generation process is diagrammed in Figure 4, where the dependence of  $\theta^{\text{PIXEL}}$  on  $d_i$  and  $v_i$  is also represented.

## 4 Learning

We use the EM algorithm (Dempster et al., 1977) to find the maximum-likelihood font parameters:  $\phi_c$ ,  $\theta_c^{\text{LPAD}}$ ,  $\theta_c^{\text{GLYPH}}$ , and  $\theta_c^{\text{RPAD}}$ . The image  $X$  is the only observed random variable in our model. The identities of the characters  $E$  the typesetting layout  $T$  and the inking  $R$  will all be unobserved. We do not learn  $\theta^{\text{INK}}$  and  $\theta^{\text{VERT}}$ , which are set to the uniform distribution.

### 4.1 Expectation Maximization

During the E-step we compute expected counts for  $E$  and  $T$ , but maximize over  $R$ , for which

we compute hard counts. Our model is an instance of a hidden semi-Markov model (HSMM), and therefore the computation of marginals is tractable with the semi-Markov forward-backward algorithm (Levinson, 1986).

During the M-step, we update the parameters  $\theta_c^{\text{LPAD}}$ ,  $\theta_c^{\text{RPAD}}$  using the standard closed-form multinomial updates and use a specialized closed-form update for  $\theta_c^{\text{GLYPH}}$  that enforces unimodality of the glyph width distribution.<sup>3</sup> The glyph weights,  $\phi_c$ , do not have a closed-form update. The noise model that  $\phi_c$  parameterizes is a local log-linear model, so we follow the approach of Berg-Kirkpatrick et al. (2010) and use L-BFGS (Liu and Nocedal, 1989) to optimize the expected likelihood with respect to  $\phi_c$ .

### 4.2 Coarse-to-Fine Learning and Inference

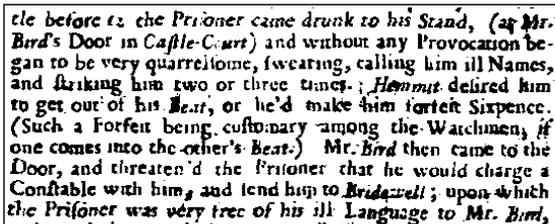
The number of states in the dynamic programming lattice grows exponentially with the order of the language model (Jelinek, 1998; Koehn, 2004). As a result, inference can become slow when the language model order  $n$  is large. To remedy this, we take a coarse-to-fine approach to both learning and inference. On each iteration of EM, we perform two passes: a coarse pass using a low-order language model, and a fine pass using a high-order language model (Petrov et al., 2008; Zhang and Gildea, 2008). We use the marginals<sup>4</sup> from the coarse pass to prune states from the dynamic program of the fine pass.

In the early iterations of EM, our font parameters are still inaccurate, and to prune heavily based on such parameters would rule out correct analyses. Therefore, we gradually increase the aggressiveness of pruning over the course of EM. To ensure that each iteration takes approximately the same amount of computation, we also gradually increase the order of the fine pass, only reaching the full order  $n$  on the last iteration. To produce a decoding of the image into text, on the final iteration we run a Viterbi pass using the pruned fine model.

<sup>3</sup>We compute the weighted mean and weighted variance of the glyph width expected counts. We set  $\theta_c^{\text{GLYPH}}$  to be proportional to a discretized Gaussian with the computed mean and variance. This update is approximate in the sense that it does not necessarily find the unimodal multinomial that maximizes expected log-likelihood, but it works well in practice.

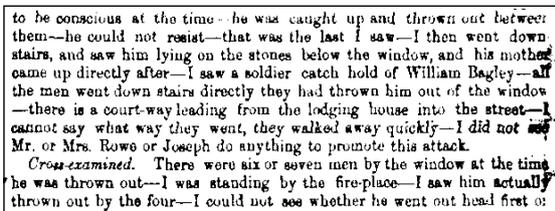
<sup>4</sup>In practice, we use max-marginals for pruning to ensure that there is still a valid path in the pruned lattice.

(a) Old Bailey, 1725:



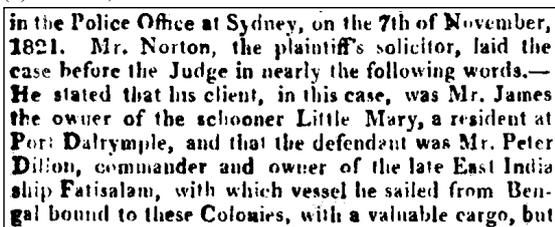
tle before the Prisoner came drunk to his Stand, (at Mr. Bird's Door in Castle-Court) and without any Provocation began to be very quarrelsome, swearing, calling him ill Names, and striking him two or three times; Hemmit delivred him to get out of his Beat, or he'd make him forfeit Sixpence. (Such a Forfeit being customary among the Watchmen, if one comes into the other's Beat.) Mr. Bird then came to the Door, and threaten'd the Prisoner that he would charge a Constable with him, and lend him to Bridewell; upon which the Prisoner was very free of his ill Language to Mr. Bird,

(b) Old Bailey, 1875:



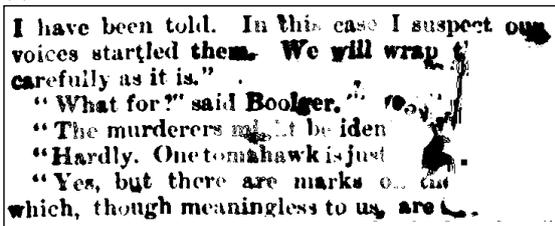
to be conscious at the time—he was caught up and thrown out between them—he could not resist—that was the last I saw—I then went down stairs, and saw him lying on the stones below the window, and his mother came up directly after—I saw a soldier catch hold of William Bagley—all the men went down stairs directly they had thrown him out of the window—there is a court-way leading from the lodging house into the street—I cannot say what way they went, they walked away quickly—I did not see Mr. or Mrs. Rowe or Joseph do anything to promote this attack.  
*Cross-examined.* There were six or seven men by the window at the time he was thrown out—I was standing by the fire-place—I saw him actually thrown out by the four—I could not see whether he went out head first or

(c) Trove, 1823:



in the Police Office at Sydney, on the 7th of November, 1821. Mr. Norton, the plaintiff's solicitor, laid the case before the Judge in nearly the following words.— He stated that his client, in this case, was Mr. James the owner of the schooner Little Mary, a resident at Port Dalrymple, and that the defendant was Mr. Peter Dillon, commander and owner of the late East India ship Fatissalam, with which vessel he sailed from Bengal bound to these Colonies, with a valuable cargo, but

(d) Trove, 1883:



I have been told. In this case I suspect our voices startled them. We will wrap it carefully as it is."  
"What for?" said Boolger.  
"The murderers must be iden  
"Hardly. One tomahawk is just  
"Yes, but there are marks on the  
which, though meaningless to us, are

Figure 6: Portions of several documents from our test set representing a range of difficulties are displayed. On document (a), which exhibits noisy typesetting, our system achieves a word error rate (WER) of 25.2. Document (b) is cleaner in comparison, and on it we achieve a WER of 15.4. On document (c), which is also relatively clean, we achieve a WER of 12.5. On document (d), which is severely degraded, we achieve a WER of 70.0.

## 5 Data

We perform experiments on two historical datasets consisting of images of documents printed between 1700 and 1900 in England and Australia. Examples from both datasets are displayed in Figure 6.

### 5.1 Old Bailey

The first dataset comes from a large set of images of the proceedings of the Old Bailey, a criminal court in London, England (Shoemaker, 2005). The Old Bailey curatorial effort, after deciding that current OCR systems do not adequately handle 18th century fonts, manually transcribed the

documents into text. We will use these manual transcriptions to evaluate the output of our system. From the Old Bailey proceedings, we extracted a set of 20 images, each consisting of 30 lines of text to use as our first test set. We picked 20 documents, printed in consecutive decades. The first document is from 1715 and the last is from 1905. We choose the first document in each of the corresponding years, choose a random page in the document, and extracted an image of the first 30 consecutive lines of text consisting of full sentences.<sup>5</sup> The ten documents in the Old Bailey dataset that were printed before 1810 use the long s glyph, while the remaining ten do not.

### 5.2 Trove

Our second dataset is taken from a collection of digitized Australian newspapers that were printed between the years of 1803 and 1954. This collection is called Trove, and is maintained by the the National Library of Australia (Holley, 2010). We extracted ten images from this collection in the same way that we extracted images from Old Bailey, but starting from the year 1803. We manually produced our own gold annotations for these ten images. Only the first document of Trove uses the long s glyph.

### 5.3 Pre-processing

Many of the images in historical collections are bitonal (binary) as a result of how they were captured on microfilm for storage in the 1980s (Arlitsch and Herbert, 2004). This is part of the reason our model is designed to work directly with binarized images. For consistency, we binarized the images in our test sets that were not already binary by thresholding pixel values.

Our model requires that the image be pre-segmented into lines of text. We automatically segment lines by training an HSMM over rows of pixels. After the lines are segmented, each line is resampled so that its vertical resolution is 30 pixels. The line extraction process also identifies pixels that are not located in central text regions, and are part of large connected components of ink, spanning multiple lines. The values of such pixels are treated as unobserved in the model since, more often than not, they are part of ink blotches.

<sup>5</sup>This ruled out portions of the document with extreme structural abnormalities, like title pages and lists. These might be interesting to model, but are not within the scope of this paper.

## 6 Experiments

We evaluate our system by comparing our text recognition accuracy to that of two state-of-the-art systems.

### 6.1 Baselines

Our first baseline is Google’s open source OCR system, Tesseract (Smith, 2007). Tesseract takes a pipelined approach to recognition. Before recognizing the text, the document is broken into lines, and each line is segmented into words. Then, Tesseract uses a classifier, aided by a word-unigram language model, to recognize whole words.

Our second baseline, ABBYY FineReader 11 Professional Edition,<sup>6</sup> is a state-of-the-art commercial OCR system. It is the OCR system that the National Library of Australia used to recognize the historical documents in Trove (Holley, 2010).

### 6.2 Evaluation

We evaluate the output of our system and the baseline systems using two metrics: character error rate (CER) and word error rate (WER). Both these metrics are based on edit distance. CER is the edit distance between the predicted and gold transcriptions of the document, divided by the number of characters in the gold transcription. WER is the word-level edit distance (words, instead of characters, are treated as tokens) between predicted and gold transcriptions, divided by the number of words in the gold transcription. When computing WER, text is tokenized into words by splitting on whitespace.

### 6.3 Language Model

We ran experiments using two different language models. The first language model was trained on the initial one million sentences of the New York Times (NYT) portion of the Gigaword corpus (Graff et al., 2007), which contains about 36 million words. This language model is out of domain for our experimental documents. To investigate the effects of using an in domain language model, we created a corpus composed of the manual annotations of all the documents in the Old Bailey proceedings, excluding those used in our test set. This corpus consists of approximately 32 million words. In all experiments we used a character  $n$ -gram order of six for the final Viterbi de-

<sup>6</sup><http://www.abbyy.com>

System	CER	WER
<b>Old Bailey</b>		
Google Tesseract	29.6	54.8
ABBYY FineReader	15.1	40.0
Ocular w/ NYT (this work)	12.6	28.1
Ocular w/ OB (this work)	<b>9.7</b>	<b>24.1</b>
<b>Trove</b>		
Google Tesseract	37.5	59.3
ABBYY FineReader	22.9	49.2
Ocular w/ NYT (this work)	<b>14.9</b>	<b>33.0</b>

Table 1: We evaluate the predicted transcriptions in terms of both character error rate (CER) and word error rate (WER), and report macro-averages across documents. We compare with two baseline systems: Google’s open source OCR system, Tesseract, and a state-of-the-art commercial system, ABBYY FineReader. We refer to our system as Ocular w/ NYT and Ocular w/ OB, depending on whether NYT or Old Bailey is used to train the language model.

coding pass and an order of three for all coarse passes.

### 6.4 Initialization and Tuning

We used as a development set ten additional documents from the Old Bailey proceedings and five additional documents from Trove that were not part of our test set. On this data, we tuned the model’s hyperparameters<sup>7</sup> and the parameters of the pruning schedule for our coarse-to-fine approach.

In experiments we initialized  $\theta_c^{\text{RPAD}}$  and  $\theta_c^{\text{LPAD}}$  to be uniform, and initialized  $\theta_c^{\text{GLYPH}}$  and  $\phi_c$  based on the standard modern fonts included with the Ubuntu Linux 12.04 distribution.<sup>8</sup> For documents that use the long  $s$  glyph, we introduce a special character type for the non-word-final  $s$ , and initialize its parameters from a mixture of the modern  $f$  and  $|$  glyphs.<sup>9</sup>

## 7 Results and Analysis

The results of our experiments are summarized in Table 1. We refer to our system as Ocular w/ NYT or Ocular w/ OB, depending on whether the language model was trained using NYT or Old Bailey, respectively. We compute macro-averages

<sup>7</sup>One of the hyperparameters we tune is the exponent of the language model. This balances the contributions of the language model and the typesetting model to the posterior (Och and Ney, 2004).

<sup>8</sup><http://www.ubuntu.com/>

<sup>9</sup>Following Berg-Kirkpatrick et al. (2010), we use a regularization term in the optimization of the log-linear model parameters  $\phi_c$  during the M-step. Instead of regularizing towards zero, we regularize towards the initializer. This slightly improves performance on our development set and can be thought of as placing a prior on the glyph shape parameters.

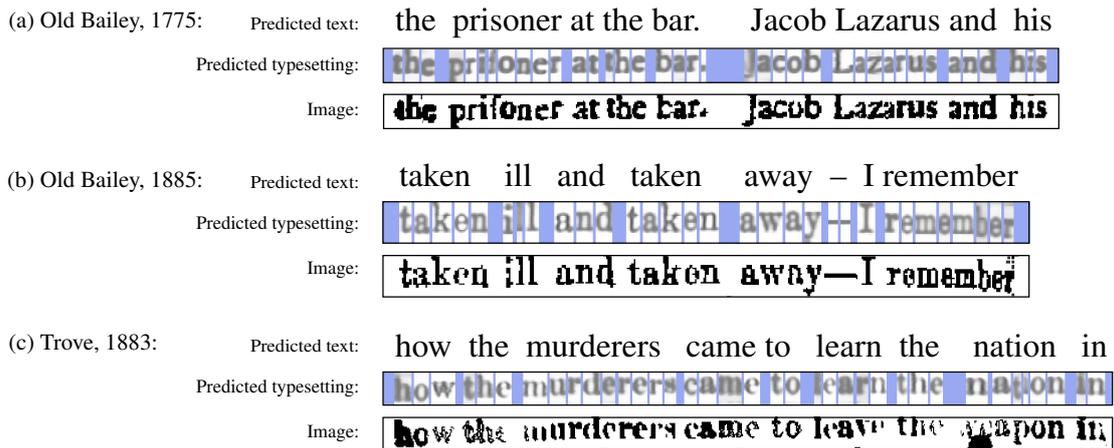


Figure 7: For each of these portions of test documents, the first line shows the transcription predicted by our model and the second line shows a representation of the learned typesetting layout. The grayscale glyphs show the Bernoulli pixel distributions learned by our model, while the padding regions are depicted in blue. The third line shows the input image.

across documents from all years. Our system, using the NYT language model, achieves an average WER of 28.1 on Old Bailey and an average WER of 33.0 on Trove. This represents a substantial error reduction compared to both baseline systems.

If we average over the documents in both Old Bailey and Trove, we find that Tesseract achieved an average WER of 56.3, ABBYY FineReader achieved an average WER of 43.1, and our system, using the NYT language model, achieved an average WER of 29.7. This means that while Tesseract incorrectly predicts more than half of the words in these documents, our system gets more than three-quarters of them right. Overall, we achieve a relative reduction in WER of 47% compared to Tesseract and 31% compared to ABBYY FineReader.

The baseline systems do not have special provisions for the long  $s$  glyph. In order to make sure the comparison is fair, we separately computed average WER on only the documents from after 1810 (which do not use the long  $s$  glyph). We found that using this evaluation our system actually achieves a larger relative reduction in WER: 50% compared to Tesseract and 35% compared to ABBYY FineReader.

Finally, if we train the language model using the Old Bailey corpus instead of the NYT corpus, we see an average improvement of 4 WER on the Old Bailey test set. This means that the domain of the language model is important, but, the results are not affected drastically even when using a language model based on modern corpora (NYT).

### 7.1 Learned Typesetting Layout

Figure 7 shows a representation of the typesetting layout learned by our model for portions of several

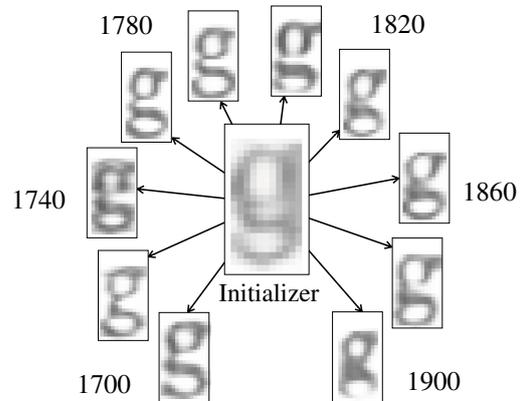


Figure 8: The central glyph is a representation of the initial model parameters for the glyph shape for  $g$ , and surrounding this are the learned parameters for documents from various years.

test documents. For each portion of a test document, the first line shows the transcription predicted by our model, and the second line shows padding and glyph regions predicted by the model, where the grayscale glyphs represent the learned Bernoulli parameters for each pixel. The third line shows the input image.

Figure 7a demonstrates a case where our model has effectively explained both the uneven baseline and over-inked glyphs by using the vertical offsets  $v_i$  and inking variables  $d_i$ . In Figure 7b the model has used glyph widths  $g_i$  and vertical offsets to explain the thinning of glyphs and falling baseline that occurred near the binding of the book. In separate experiments on the Old Bailey test set, using the NYT language model, we found that removing the vertical offset variables from the model increased WER by 22, and removing the inking variables increased WER by 16. This indicates that it is very important to model both these aspects of printing press rendering.

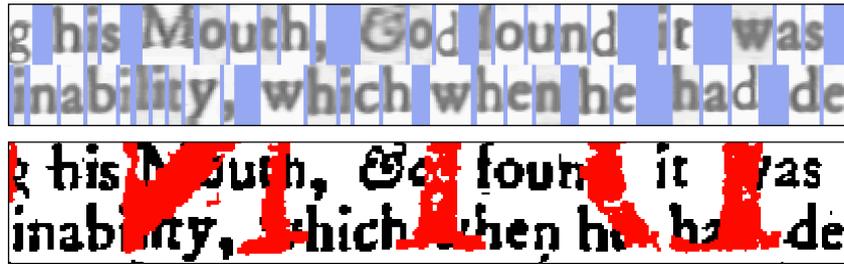


Figure 9: This Old Bailey document from 1719 has severe ink bleeding from the facing page. We annotated these blotches (in red) and treated the corresponding pixels as unobserved in the model. The layout shown is predicted by the model.

Figure 7c shows the output of our system on a difficult document. Here, missing characters and ink blotches confuse the model, which picks something that is reasonable according to the language model, but incorrect.

## 7.2 Learned Fonts

It is interesting to look at the fonts learned by our system, and track how historical fonts changed over time. Figure 8 shows several grayscale images representing the Bernoulli pixel probabilities for the most likely width of the glyph for *g* under various conditions. At the center is the representation of the initial parameter values, and surrounding this are the learned parameters for documents from various years. The learned shapes are visibly different from the initializer, which is essentially an average of modern fonts, and also vary across decades.

We can ask to what extent learning the font structure actually improved our performance. If we turn off learning and just use the initial parameters to decode, WER increases by 8 on the Old Bailey test set when using the NYT language model.

## 7.3 Unobserved Ink Blotches

As noted earlier, one strength of our generative model is that we can make the values of certain pixels unobserved in the model, and let inference fill them in. We conducted an additional experiment on a document from the Old Bailey proceedings that was printed in 1719. This document, a fragment of which is shown in Figure 9, has severe ink bleeding from the facing page. We manually annotated the ink blotches (shown in red), and made them unobserved in the model. The resulting typesetting layout learned by the model is also shown in Figure 9. The model correctly predicted most of the obscured words. Running the model with the manually specified unobserved pixels re-

duced the WER on this document from 58 to 19 when using the NYT language model.

## 7.4 Remaining Errors

We performed error analysis on our development set by randomly choosing 100 word errors from the WER alignment and manually annotating them with relevant features. Specifically, for each word error we recorded whether or not the error contained punctuation (either in the predicted word or the gold word), whether the text in the corresponding portion of the original image was italicized, and whether the corresponding portion of the image exhibited over-inking, missing ink, or significant ink blotches. These last three feature types are subjective in nature but may still be informative. We found that 56% of errors were accompanied by over-inking, 50% of errors were accompanied by ink blotches, 42% of errors contained punctuation, 21% of errors showed missing ink, and 12% of errors contained text that was italicized in the original image.

Our own subjective assessment indicates that many of these error features are in fact causal. More often than not, italicized text is incorrectly transcribed. In cases of extreme ink blotching, or large areas of missing ink, the system usually makes an error.

## 8 Conclusion

We have demonstrated a model, based on the historical typesetting process, that effectively learns font structure in an unsupervised fashion to improve transcription of historical documents into text. The parameters of the learned fonts are interpretable, as are the predicted typesetting layouts. Our system achieves state-of-the-art results, significantly outperforming two state-of-the-art baseline systems.

## References

- Kenning Arlitsch and John Herbert. 2004. Microfilm, paper, and OCR: Issues in newspaper digitization. the Utah digital newspapers program. *Microform & Imaging Review*.
- Taylor Berg-Kirkpatrick and Dan Klein. 2011. Simple effective decipherment via combinatorial optimization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Arthur Dempster, Nan Laird, and Donald Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*.
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English Gigaword third edition. Linguistic Data Consortium, Catalog Number LDC2007T07.
- Tin Kam Ho and George Nagy. 2000. OCR with no shape training. In *Proceedings of the 15th International Conference on Pattern Recognition*.
- Rose Holley. 2010. Trove: Innovation in access to information in Australia. *Ariadne*.
- Gary Huang, Erik G Learned-Miller, and Andrew McCallum. 2006. Cryptogram decoding for optical character recognition. *University of Massachusetts-Amherst Technical Report*.
- Fred Jelinek. 1998. *Statistical methods for speech recognition*. MIT press.
- Andrew Kae and Erik Learned-Miller. 2009. Learning on the fly: font-free approaches to difficult OCR problems. In *Proceedings of the 2009 International Conference on Document Analysis and Recognition*.
- Andrew Kae, Gary Huang, Carl Doersch, and Erik Learned-Miller. 2010. Improving state-of-the-art OCR through high-precision document-specific modeling. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*.
- Vladimir Kluzner, Asaf Tzadok, Yuval Shimony, Eugene Walach, and Apostolos Antonacopoulos. 2009. Word-based adaptive OCR for historical books. In *Proceedings of the 2009 International Conference on Document Analysis and Recognition*.
- Vladimir Kluzner, Asaf Tzadok, Dan Chevion, and Eugene Walach. 2011. Hybrid approach to adaptive OCR for historical books. In *Proceedings of the 2011 International Conference on Document Analysis and Recognition*.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*.
- Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. *Machine translation: From real users to research*.
- Okan Kolak, William Byrne, and Philip Resnik. 2003. A generative probabilistic OCR model for NLP applications. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Gary Kopec and Mauricio Lomelin. 1996. Document-specific character template estimation. In *Proceedings of the International Society for Optics and Photonics*.
- Gary Kopec, Maya Said, and Kris Popat. 2001. N-gram language models for document image decoding. In *Proceedings of Society of Photographic Instrumentation Engineers*.
- Stephen Levinson. 1986. Continuously variable duration hidden Markov models for automatic speech recognition. *Computer Speech & Language*.
- Dong C Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming*.
- Franz Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*.
- Slav Petrov, Aria Haghighi, and Dan Klein. 2008. Coarse-to-fine syntactic machine translation using language projections. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.
- Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.
- Sujith Ravi and Kevin Knight. 2011. Bayesian inference for Zodiac and other homophonic ciphers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Robert Shoemaker. 2005. Digital London: Creating a searchable web of interlinked sources on eighteenth century London. *Electronic Library and Information Systems*.
- Ray Smith. 2007. An overview of the tesseract ocr engine. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition*.

Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Georgios Vamvakas, Basilios Gatos, Nikolaos Stamatopoulos, and Stavros Perantonis. 2008. A complete optical character recognition methodology for historical documents. In *The Eighth IAPR International Workshop on Document Analysis Systems*.

Hao Zhang and Daniel Gildea. 2008. Efficient multi-pass decoding for synchronous context free grammars. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.

# Adapting Discriminative Reranking to Grounded Language Learning

**Joohyun Kim**

Department of Computer Science  
The University of Texas at Austin  
Austin, TX 78701, USA  
scimitar@cs.utexas.edu

**Raymond J. Mooney**

Department of Computer Science  
The University of Texas at Austin  
Austin, TX 78701, USA  
mooney@cs.utexas.edu

## Abstract

We adapt discriminative reranking to improve the performance of grounded language acquisition, specifically the task of learning to follow navigation instructions from observation. Unlike conventional reranking used in syntactic and semantic parsing, gold-standard reference trees are not naturally available in a grounded setting. Instead, we show how the weak supervision of response feedback (e.g. successful task completion) can be used as an alternative, experimentally demonstrating that its performance is comparable to training on gold-standard parse trees.

## 1 Introduction

Grounded language acquisition involves learning to comprehend and/or generate language by simply observing its use in a naturally occurring context in which the meaning of a sentence is grounded in perception and/or action (Roy, 2002; Yu and Ballard, 2004; Gold and Scassellati, 2007; Chen et al., 2010). Börschinger et al. (2011) introduced an approach that reduces grounded language learning to unsupervised *probabilistic context-free grammar* (PCFG) induction and demonstrated its effectiveness on the task of sportscasting simulated robot soccer games. Subsequently, Kim and Mooney (2012) extended their approach to make it tractable for the more complex problem of learning to follow natural-language navigation instructions from observations of humans following such instructions in a virtual environment (Chen and Mooney, 2011). The observed sequence of actions provides very weak, ambiguous supervision for learning instructional language since there are many possible ways to describe the same execution path. Although their approach improved accuracy on the navigation task compared

to the original work of Chen and Mooney (2011), it was still far from human performance.

Since their system employs a generative model, *discriminative reranking* (Collins, 2000) could potentially improve its performance. By training a discriminative classifier that uses global features of complete parses to identify correct interpretations, a reranker can significantly improve the accuracy of a generative model. Reranking has been successfully employed to improve syntactic parsing (Collins, 2002b), semantic parsing (Lu et al., 2008; Ge and Mooney, 2006), semantic role labeling (Toutanova et al., 2005), and named entity recognition (Collins, 2002c). Standard reranking requires gold-standard interpretations (e.g. parse trees) to train the discriminative classifier. However, grounded language learning does not provide gold-standard interpretations for the training examples. Only the ambiguous perceptual context of the utterance is provided as supervision. For the navigation task, this supervision consists of the observed sequence of actions taken by a human when following an instruction. Therefore, it is impossible to directly apply conventional discriminative reranking to such problems. We show how to adapt reranking to work with such weak supervision. Instead of using gold-standard annotations to determine the correct interpretations, we simply prefer interpretations of navigation instructions that, when executed in the world, actually reach the intended destination. Additionally, we extensively revise the features typically used in parse reranking to work with the PCFG approach to grounded language learning.

The rest of the paper is organized as follows: Section 2 reviews the navigation task and the PCFG approach to grounded language learning. Section 3 presents our modified approach to reranking and Section 4 describes the novel features used to evaluate parses. Section 5 experimentally evaluates the approach comparing to sev-





est execution success. This requires an external module that evaluates the execution accuracy of the candidate parses. For the navigation task, we use the MARCO (MacMahon et al., 2006) execution module, which is also used to evaluate how well the overall system learns to follow directions (Chen and Mooney, 2011). Since MARCO is nondeterministic when executing underspecified plans, we execute each candidate plan 10 times, and its execution rate is the percentage of trials in which it reaches the correct destination. When there are multiple candidate parses tied for the highest execution rate, the one assigned the largest probability by the baseline model is selected. Our modified averaged perceptron procedure with such a response-based update is shown in Algorithm 1.

One additional issue must be addressed when computing the output of the GEN function. The final plan MRs are produced from parse trees using compositional semantics (see Kim and Mooney (2012) for details). Consequently, the  $n$ -best parse trees for the baseline model do not necessarily produce the  $n$ -best *distinct* plans, since many parses can produce the same plan. Therefore, we adapt the GEN function to produce the  $n$  best distinct plans rather than the  $n$  best parses. This may require examining many more than the  $n$  best parses, because many parses have insignificant differences that do not affect the final plan. The score assigned to a plan is the probability of the most probable parse that generates that plan. In order to efficiently compute the  $n$  best plans, we modify the exact  $n$ -best parsing algorithm developed by Huang and Chiang (2005). The modified algorithm ensures that each plan in the computed  $n$  best list produces a new distinct plan.

### 3.2 Weight Updates Using Multiple Parses

Typically, when used for reranking, the averaged perceptron updates its weights using the feature-vector difference between the current best predicted candidate and the gold-standard reference (line 6 in Algorithm 1). In our initial modified version, we replaced the gold-standard reference parse with the pseudo-gold reference, which has the highest execution rate amongst all candidate parses. However, this ignores all other candidate parses during perceptron training. However, it is not ideal to regard other candidate parses as “useless.” There may be multiple candidate parses with the same maximum execution rate, and even can-

didates with lower execution rates could represent the correct plan for the instruction given the weak, indirect supervision provided by the observed sequence of human actions.

Therefore, we also consider a further modification of the averaged perceptron algorithm which updates its weights using multiple candidate parses. Instead of only updating the weights with the single difference between the predicted and pseudo-gold parses, the weight vector  $\bar{W}$  is updated with the sum of feature-vector differences between the current predicted candidate and *all* other candidates that have a higher execution rate. Formally, in this version, we replace lines 5–6 of Algorithm 1 with:

- 1: **for all**  $y \in \text{GEN}(e_i)$  where  $y \neq y_i$  and  $\text{EXEC}(y) > \text{EXEC}(y_i)$  **do**
- 2:      $\bar{W} = \bar{W} + (\text{EXEC}(y) - \text{EXEC}(y_i)) \times (\Phi(e_i, y) - \Phi(e_i, y_i))$
- 3: **end for**

where  $\text{EXEC}(y)$  is the execution rate of the MR plan  $m$  derived from parse tree  $y$ .

In the experiments below, we demonstrate that, by exploiting multiple reference parses, this new update rule increases the execution accuracy of the final system. Intuitively, this approach gathers additional information from all candidate parses with higher execution accuracy when learning the discriminative reranker. In addition, as shown in line 2 of the algorithm above, it uses the difference in execution rates between a candidate and the currently preferred parse to weight the update to the parameters for that candidate. This allows more effective plans to have a larger impact on the learned model in each iteration.

## 4 Reranking Features

This section describes the features  $\Phi$  extracted from parses produced by the generative model and used to rerank the candidates.

### 4.1 Base Features

The base features adapt those used in previous reranking methods, specifically those of Collins (2002a), Lu et al. (2008), and Ge and Mooney (2006), which are directly extracted from parse trees. In addition, we also include the log probability of the parse tree as an additional feature. Figure 3 shows a sample full parse tree from our baseline model, which is used when explaining the

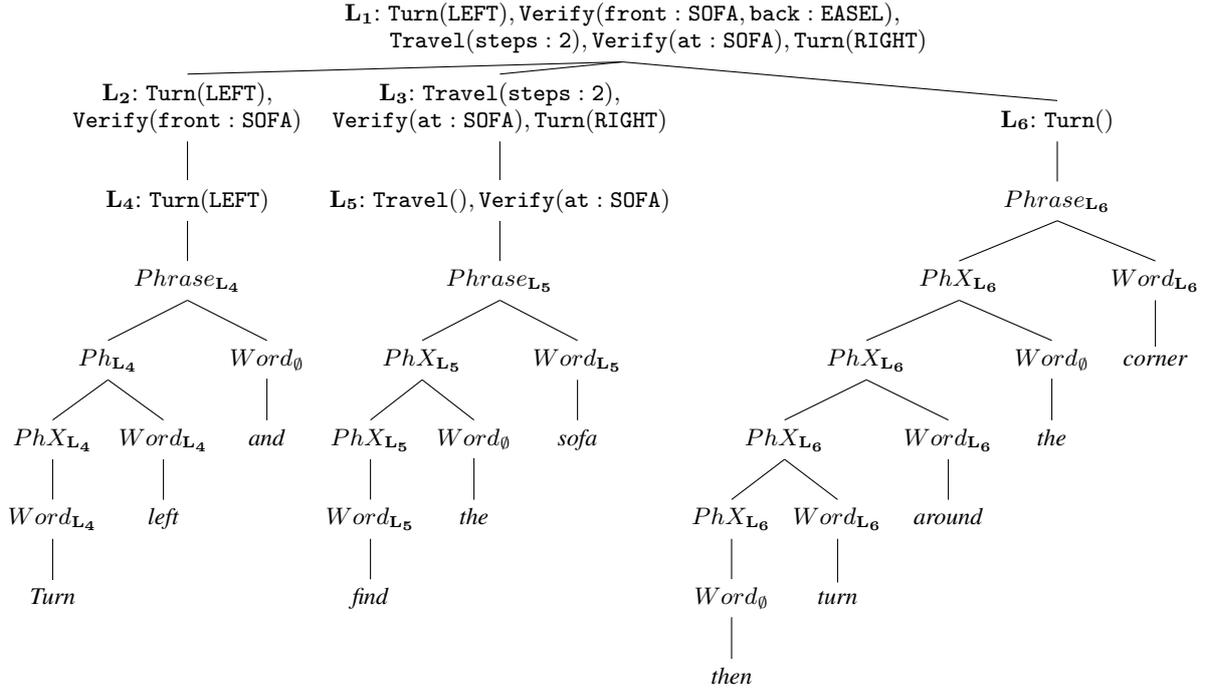


Figure 3: Sample full parse tree for the sentence “Turn left and find the sofa then turn around the corner” used to explain reranking features. Nonterminals representing MR plan components are shown, which are labeled  $L_1$  to  $L_6$  for ease of reference. Additional nonterminals such as *Phrase*, *Ph*, *PhX*, and *Word* are subsidiary ones for generating NL words from MR nonterminals. They are also shown in order to represent the entire process of how parse trees are constructed (for details, refer to Kim and Mooney (2012)).

reranking features below, each illustrated by an example.

- a) PCFG Rule. Indicates whether a particular PCFG rule is used in the parse tree:  $f(L_1 \Rightarrow L_2 L_3) = 1$ .
- b) Grandparent PCFG Rule. Indicates whether a particular PCFG rule *as well as* the nonterminal above it is used in the parse tree:  $f(L_3 \Rightarrow L_5 L_6 | L_1) = 1$ .
- c) Long-range Unigram. Indicates whether a nonterminal has a given NL word below it in the parse tree:  $f(L_2 \rightsquigarrow \text{left}) = 1$  and  $f(L_4 \rightsquigarrow \text{turn}) = 1$ .
- d) Two-level Long-range Unigram. Indicates whether a nonterminal has a child nonterminal which eventually generates a NL word in the parse tree:  $f(L_4 \rightsquigarrow \text{left} | L_2) = 1$
- e) Unigram. Indicates whether a nonterminal produces a given child nonterminal or terminal NL word in the parse tree:  $f(L_1 \rightarrow L_2) = 1$  and  $f(L_1 \rightarrow L_3) = 1$ .

- f) Grandparent Unigram. Indicates whether a nonterminal has a given child nonterminal/terminal below it, as well as a given parent nonterminal:  $f(L_2 \rightarrow L_4 | L_1) = 1$
- g) Bigram. Indicates whether a given bigram of nonterminal/terminals occurs for given a parent nonterminal:  $f(L_1 \rightarrow L_2 : L_3) = 1$ .
- h) Grandparent Bigram. Same as Bigram, but also includes the nonterminal above the parent nonterminal:  $f(L_3 \rightarrow L_5 : L_6 | L_1) = 1$ .
- i) Log-probability of Parse Tree. Certainty assigned by the base generative model.

## 4.2 Predicate-Only Features

The base features above generally include nonterminal symbols used in the parse tree. In the grounded PCFG model, nonterminals are named after components of the semantic representations (MRs), which are complex and numerous. There are  $\simeq 2,500$  nonterminals in the grammar constructed for the navigation data, most of which are very specific and rare. This results in a very large, sparse feature space which can easily lead

the reranking model to over-fit the training data and prevent it from generalizing properly.

Therefore, we also tried constructing more general features that are less sparse. First, we construct generalized versions of the base features in which nonterminal symbols use only predicate names and omit their arguments. In the navigation task, action arguments frequently contain redundant, rarely used information. In particular, the interleaving verification steps frequently include many details that are never actually mentioned in the NL instructions. For instance, a nonterminal for the MR

```
Turn(LEFT),
Verify(at:SOFA, front:EASEL),
Travel(steps:3)
```

is transformed into the predicate-only form

```
Turn(), Verify(), Travel()
```

, and then used to construct more general versions of the base features described in the previous section. Second, another version of the base features are constructed in which nonterminal symbols include action arguments but omit all interleaving verification steps. This is a somewhat more conservative simplification of the nonterminal symbols. Although verification steps sometimes help interpret the actions and their surrounding context, they frequently cause the nonterminal symbols to become unnecessarily complex and specific.

### 4.3 Descended Action Features

Finally, another feature group which we utilize captures whether a particular atomic action in a nonterminal “descends” into one of its child nonterminals or not. An atomic action consists of a predicate and its arguments, e.g. `Turn(LEFT)`, `Travel(steps:2)`, or `Verify(at:SOFA)`. When an atomic action descends into lower nonterminals in a parse tree, it indicates that it is mentioned in the NL instruction and is therefore important. Below are several feature types related to descended actions that are used in our reranking model:

- a) **Descended Action.** Indicates whether a given atomic action in a nonterminal descends to the next level. In Figure 3,  $f(\text{Turn}(\text{LEFT})) = 1$  since it descends into  $L_2$  and  $L_4$ .
- b) **Descended Action Unigram.** Same as Descended Action, but also includes the current nonterminal:  $f(\text{Turn}(\text{LEFT})|L_1) = 1$ .

- c) **Grandparent Descended Action Unigram.** Same as Descended Action Unigram, but additionally includes the parent nonterminal as well as the current one:  $f(\text{Turn}(\text{LEFT})|L_2, L_1) = 1$ .

- d) **Long-range Descended Action Unigram.** Indicates whether a given atomic action in a nonterminal descends to a child nonterminal and this child generates a given NL word below it:  $f(\text{Turn}(\text{LEFT}) \rightsquigarrow \text{left}) = 1$

## 5 Experimental Evaluation

### 5.1 Data and Methodology

The navigation data was collected by MacMahon et al. (2006), and includes English instructions and human follower data.<sup>1</sup> The data contains 706 route instructions for three virtual worlds. The instructions were produced by six instructors for 126 unique starting and ending location pairs over the three maps. Each instruction is annotated with 1 to 15 human follower traces with an average of 10.4 actions per instruction. Each instruction contains an average of 5.0 sentences each with an average of 7.8 words. Chen and Mooney (2011) constructed a version of the data in which each sentence is annotated with the actions taken by the majority of followers when responding to this sentence. This single-sentence version is used for training. Manually annotated “gold standard” formal plans for each sentence are used for evaluation purposes only.

We followed the same experimental methodology as Kim and Mooney (2012) and Chen and Mooney (2011). We performed “leave one environment out” cross-validation, i.e. 3 trials of training on two environments and testing on the third. The baseline model is first trained on data for two environments and then used to generate the  $n = 50$  best plans for both training and testing instructions. As mentioned in Section 3.1, we need to generate many more top parse trees to get 50 distinct formal MR plans. We limit the number of best parse trees to 1,000,000, and even with this high limit, some training examples were left with less than 50 distinct plans.<sup>2</sup> Each candidate

<sup>1</sup>Data is available at <http://www.cs.utexas.edu/users/ml/clamp/navigation/>

<sup>2</sup>9.6% of the examples (310 out of total 3237) produced less than 50 distinct MR plans in the evaluation. This was mostly due to exceeding the parse-tree limit and partly because the baseline model failed to parse some NL sentences.

n		1	2	5	10	25	50
Parse Accuracy	F1	74.81	79.08	82.78	85.32	87.52	88.62
Plan Execution	Single-sentence	57.22	63.86	70.93	76.41	83.59	87.02
	Paragraph	20.17	28.08	35.34	40.64	48.69	53.66

Table 1: Oracle parse and execution accuracy for single sentence and complete paragraph instructions for the  $n$  best parses.

plan is then executed using MARCO and its rate of successfully reaching the goal is recorded. Our reranking model is then trained on the training data using the  $n$ -best candidate parses. We only retain reranking features that appear (i.e. have a value of 1) at least twice in the training data.

Finally, we measure both parse and execution accuracy on the test data. Parse accuracy evaluates how well a system maps novel NL sentences for new environments into correct MR plans (Chen and Mooney, 2011). It is calculated by comparing the system’s MR output to the gold-standard MR. Accuracy is measured using F1, the harmonic mean of precision and recall for individual MR constituents, thereby giving partial credit to approximately correct MRs. We then execute the resulting MR plans in the test environment to see whether they successfully reach the desired destinations. Execution is evaluated both for single sentence and complete paragraph instructions. Successful execution rates are calculated by averaging 10 nondeterministic MARCO executions.

## 5.2 Reranking Results

### Oracle results

As typical in reranking experiments, we first present results for an “oracle” that always returns the best result amongst the top- $n$  candidates produced by the baseline system, thereby providing an upper bound on the improvements possible with reranking. Table 1 shows oracle accuracy for both semantic parsing and plan execution for single sentence and complete paragraph instructions for various values of  $n$ . For oracle parse accuracy, for each sentence, we pick the parse that gives the highest F1 score. For oracle single-sentence execution accuracy, we pick the parse that gives the highest execution success rate. These single-sentence plans are then concatenated to produce a complete plan for each paragraph instruction in order to measure overall execution accuracy. Since making an error in *any* of the sentences in an in-

struction can easily lead to the wrong final destination, paragraph-level accuracies are always much lower than sentence-level ones. In order to balance oracle accuracy and the computational effort required to produce  $n$  distinct plans, we chose  $n = 50$  for the final experiments since oracle performance begins to asymptote at this point.

### Response-based vs. gold-standard reference weight updates

Table 2 presents reranking results for our proposed response-based weight update (*Single*) for the averaged perceptron (cf. Section 3.1) compared to the typical weight update method using gold-standard parses (*Gold*). Since the gold-standard annotation gives the correct MR rather than a parse tree for each sentence, *Gold* selects as a single reference parse the candidate in the top 50 whose resulting MR is most similar to the gold-standard MR as determined by its parse accuracy. Ge and Mooney (2006) employ a similar approach when reranking semantic parses.

The results show that our response-based approach (*Single*) has better execution accuracy than both the baseline *and* the standard approach using gold-standard parses (*Gold*). However, *Gold* does perform best on parse accuracy since it explicitly focuses on maximizing the accuracy of the resulting MR. In contrast, by focusing discriminative training on optimizing performance of the ultimate end task, our response-based approach actually outperforms the traditional approach on the final task. In addition, it only utilizes feedback that is naturally available for the task, rather than requiring an expert to laboriously annotate each sentence with a gold-standard MR. Even though *Gold* captures more elements of the gold-standard MRs, it may miss some critical MR components that are crucial to the final navigation task. The overall result is very promising because it demonstrates how reranking can be applied to grounded language learning tasks where gold-standard parses are not readily available.

	Parse Acc	Plan Execution	
	F1	Single	Para
Baseline	74.81	57.22	20.17
Gold	<b>78.26</b>	52.57	19.33
Single	73.32	59.65	22.62
Multi	73.43	<b>62.81</b>	<b>26.57</b>

Table 2: Reranking results comparing our response-based methods using single (*Single*) or multiple (*Multi*) pseudo-gold parses to the standard approach using a single gold-standard parse (*Gold*). *Baseline* refers to Kim and Mooney (2012)’s system. Reranking results use all features described in Section 4. “*Single*“ means the single-sentence version and “*Para*” means the full paragraph version of the corpus.

### Weight update with single vs. multiple reference parses

Table 2 also shows performance when using multiple reference parse trees to update weights (cf. Section 3.2). Using multiple parses (*Multi*) clearly performs better for all evaluation metrics, particularly execution. As explained in Section 3.2, the single-best pseudo-gold parse provides weak, ambiguous feedback since it only provides a rough estimate of the response feedback from the execution module. Using a variety of preferable parses to update weights provides a greater amount and variety of weak feedback and therefore leads to a more accurate model.<sup>3</sup>

### Comparison of different feature groups

Table 3 compares reranking results using the different feature groups described in Section 4. Compared to the baseline model (Kim and Mooney, 2012), each of the feature groups *Base* (base features), *Pred* (predicate-only and verification-removed features), and *Desc* (descended action features) helps improve the performance of plan execution for both single sentence and complete paragraph navigation instructions. Among them, *Desc* is the most effective group of features. Combinations of the feature groups helps fur-

<sup>3</sup>We also tried extending *Gold* to use multiple reference parses in the same manner, but this actually degraded its performance for all metrics. This indicates that, unlike *Multi*, parses other than the best one do not have useful information in terms of optimizing normal parse accuracy. Instead, additional parses seem to add noise to the training process in this case. Therefore, updating with multiple parses does not appear to be useful in standard reranking.

Features	Parse Acc	Plan Execution	
	F1	Single	Para
Baseline	<b>74.81</b>	57.22	20.17
Base	71.50	60.09	23.20
Pred	71.61	60.87	24.13
Desc	73.90	61.33	25.00
Base+Pred	69.52	61.49	26.24
Base+Desc	73.66	61.72	25.58
Pred+Desc	72.56	62.36	26.04
All	73.43	<b>62.81</b>	<b>26.57</b>

Table 3: Reranking results comparing different sets of features. *Base* refers to base features (cf. Section 4.1), *Pred* refers to predicate-only features and also includes features based on removing interleaving verification steps (cf. Section 4.2), *Desc* refers to descended action features (cf. Section 4.3). *All* refers to all the features including *Base*, *Pred*, and *Desc*. All results use weight update with multiple reference parses (cf. Section 3.2).

ther improve the plan execution performance, and reranking using all of the feature groups (*All*) performs the best, as expected. However, since our model is optimizing plan execution during training, the results for parse accuracy are always worse than the baseline model.

## 6 Related Work

Discriminative reranking is a common machine learning technique to improve the output of generative models. It has been shown to be effective for various natural language processing tasks including syntactic parsing (Collins, 2000; Collins, 2002b; Collins and Koo, 2005; Charniak and Johnson, 2005; Huang, 2008), semantic parsing (Lu et al., 2008; Ge and Mooney, 2006), part-of-speech tagging (Collins, 2002a), semantic role labeling (Toutanova et al., 2005), named entity recognition (Collins, 2002c), machine translation (Shen et al., 2004; Fraser and Marcu, 2006) and surface realization in generation (White and Rajkumar, 2009; Konstas and Lapata, 2012). However, to our knowledge, there has been no previous attempt to apply discriminative reranking to grounded language acquisition, where gold-standard reference parses are not typically available for training reranking models.

Our use of response-based training is similar

to work on learning semantic parsers from execution output such as the answers to database queries (Clarke et al., 2010; Liang et al., 2011). Although the demands of grounded language tasks, such as following navigation instructions, are different, it would be interesting to try adapting these alternative approaches to such problems.

## 7 Future Work

In the future, we would like to explore the construction of better, more-general reranking features that are less prone to over-fitting. Since typical reranking features rely on the combination and/or modification of nonterminals appearing in parse trees, for the large PCFG's produced for grounded language learning, such features are very sparse and rare. Although the current features provide a significant increase in performance, oracle results imply that an even larger benefit may be achievable.

In addition, employing other reranking methodologies, such as kernel methods (Collins, 2002b), and forest reranking exploiting a packed forest of exponentially many parse trees (Huang, 2008), is another area of future work. We also would like to apply our approach to other reranking algorithms such as SVMs (Joachims, 2002) and Max-Ent methods (Charniak and Johnson, 2005).

## 8 Conclusions

In this paper, we have shown how to adapt discriminative reranking to grounded language learning. Since typical grounded language learning problems, such as navigation instruction following, do not provide the gold-standard reference parses required by standard reranking models, we have devised a novel method for using the weaker supervision provided by response feedback (e.g. the execution of inferred navigation plans) when training a perceptron-based reranker. This approach was shown to be very effective compared to the traditional method of using gold-standard parses. In addition, since this response-based supervision is weak and ambiguous, we have also presented a method for using multiple reference parses to perform perceptron weight updates and shown a clear further improvement in end-task performance with this approach.

## Acknowledgments

We thank anonymous reviewers for their helpful comments to improve this paper. This work was funded by the NSF grant IIS-0712907 and IIS-1016312. Experiments were performed on the Mastodon Cluster, provided by NSF Grant EIA-0303609.

## References

- Benjamin Börschinger, Bevan K. Jones, and Mark Johnson. 2011. Reducing grounded learning tasks to grammatical inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pages 1416–1425, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 173–180, Ann Arbor, MI, June.
- David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-2011)*, San Francisco, CA, USA, August.
- David L. Chen, Joohyun Kim, and Raymond J. Mooney. 2010. Training a multilingual sportscaster: Using perceptual context to learn language. *Journal of Artificial Intelligence Research*, 37:397–435.
- James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world's response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL-2010)*, pages 18–27, Uppsala, Sweden, July. Association for Computational Linguistics.
- Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–69.
- Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000)*, pages 175–182, Stanford, CA, June.
- Michael Collins. 2002a. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-02)*, Philadelphia, PA, July.
- Michael Collins. 2002b. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of*

- the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002), pages 263–270, Philadelphia, PA, July.
- Michael Collins. 2002c. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 489–496, Philadelphia, PA.
- Alexander Fraser and Daniel Marcu. 2006. Semi-supervised training for statistical word alignment. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics (ACL-06)*, pages 769–776, Stroudsburg, PA, USA. Association for Computational Linguistics.
- R. Ge and R. J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, Sydney, Australia, July.
- Kevin Gold and Brian Scassellati. 2007. A robot that uses existing vocabulary to infer non-visual word meanings from observation. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 1*, AAAI’07, pages 883–888. AAAI Press.
- Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing ’05, pages 53–64, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08: HLT*, pages 586–594, Columbus, Ohio, June. Association for Computational Linguistics.
- Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002)*, Edmonton, Canada.
- Joohyun Kim and Raymond J. Mooney. 2012. Unsupervised PCFG induction for grounded language learning with highly ambiguous supervision. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Natural Language Learning*, EMNLP-CoNLL ’12.
- Ioannis Konstas and Mirella Lapata. 2012. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL ’12, pages 369–378, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of ACL*, Portland, Oregon, June. Association for Computational Linguistics.
- Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*, Honolulu, HI, October.
- Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: connecting language, knowledge, and action in route instructions. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI’06, pages 1475–1482. AAAI Press.
- Deb Roy. 2002. Learning visually grounded words and syntax for a scene description task. *Computer Speech and Language*, 16(3):353–385.
- Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 177–184, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 589–596, Ann Arbor, MI, June.
- Michael White and Rajkrishnan Rajkumar. 2009. Perceptron reranking for CCG realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP ’09, pages 410–419, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chen Yu and Dana H. Ballard. 2004. On the integration of grounding language and learning objects. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, pages 488–493.

# Universal Conceptual Cognitive Annotation (UCCA)

**Omri Abend\***

Institute of Computer Science  
The Hebrew University  
omria01@cs.huji.ac.il

**Ari Rappoport**

Institute of Computer Science  
The Hebrew University  
arir@cs.huji.ac.il

## Abstract

Syntactic structures, by their nature, reflect first and foremost the formal constructions used for expressing meanings. This renders them sensitive to formal variation both within and across languages, and limits their value to semantic applications. We present UCCA, a novel multi-layered framework for semantic representation that aims to accommodate the semantic distinctions expressed through linguistic utterances. We demonstrate UCCA's portability across domains and languages, and its relative insensitivity to meaning-preserving syntactic variation. We also show that UCCA can be effectively and quickly learned by annotators with no linguistic background, and describe the compilation of a UCCA-annotated corpus.

## 1 Introduction

Syntactic structures are mainly committed to representing the formal patterns of a language, and only indirectly reflect semantic distinctions. For instance, while virtually all syntactic annotation schemes are sensitive to the structural difference between (a) “John took a shower” and (b) “John showered”, they seldom distinguish between (a) and the markedly different (c) “John took my book”. In fact, the annotations of (a) and (c) are identical under the most widely-used schemes for English, the Penn Treebank (PTB) (Marcus et al., 1993) and CoNLL-style dependencies (Surdeanu et al., 2008) (see Figure 1).

\* Omri Abend is grateful to the Azrieli Foundation for the award of an Azrieli Fellowship.

Underscoring the semantic similarity between (a) and (b) can assist semantic applications. One example is machine translation to target languages that do not express this structural distinction (e.g., both (a) and (b) would be translated to the same German sentence “John duschte”). Question Answering applications can also benefit from distinguishing between (a) and (c), as this knowledge would help them recognize “my book” as a much more plausible answer than “a shower” to the question “what did John take?”.

This paper presents a novel approach to grammatical representation that annotates semantic distinctions and aims to abstract away from specific syntactic constructions. We call our approach *Universal Conceptual Cognitive Annotation (UCCA)*. The word “cognitive” refers to the type of categories UCCA uses and its theoretical underpinnings, and “conceptual” stands in contrast to “syntactic”. The word “universal” refers to UCCA's capability to accommodate a highly rich set of semantic distinctions, and its aim to ultimately provide all the necessary semantic information for learning grammar. In order to accommodate this rich set of distinctions, UCCA is built as a multi-layered structure, which allows for its open-ended extension. This paper focuses on the foundational layer of UCCA, a coarse-grained layer that represents some of the most important relations expressed through linguistic utterances, including argument structure of verbs, nouns and adjectives, and the inter-relations between them (Section 2).

UCCA is supported by extensive typological cross-linguistic evidence and accords with the leading Cognitive Linguistics theories. We build primarily on Basic Linguistic Theory (BLT) (Dixon, 2005; 2010a; 2010b; 2012), a typological approach to grammar successfully used for the de-

scription of a wide variety of languages. BLT uses semantic similarity as its main criterion for categorizing constructions both within and across languages. UCCA takes a similar approach, thereby creating a set of distinctions that is motivated cross-linguistically. We demonstrate UCCA’s relative insensitivity to paraphrasing and to cross-linguistic variation in Section 4.

UCCA is exceptional in (1) being a semantic scheme that abstracts away from specific syntactic forms and is not defined relative to a specific domain or language, (2) providing a coarse-grained representation which allows for open-ended extension, and (3) using cognitively-motivated categories. An extensive comparison of UCCA to existing approaches to syntactic and semantic representation, focusing on the major resources available for English, is found in Section 5.

This paper also describes the compilation of a UCCA-annotated corpus. We provide a quantitative assessment of the annotation quality. Our results show a quick learning curve and no substantial difference in the performance of annotators with and without background in linguistics. This is an advantage of UCCA over its syntactic counterparts that usually need annotators with extensive background in linguistics (see Section 3).

We note that UCCA’s approach that advocates automatic learning of syntax from semantic supervision stands in contrast to the traditional view of generative grammar (Clark and Lappin, 2010).

## 2 The UCCA Scheme

### 2.1 The Formalism

UCCA uses directed acyclic graphs (DAGs) to represent its semantic structures. The atomic meaning-bearing units are placed at the leaves of the DAG and are called *terminals*. In the foundational layer, terminals are words and multi-word chunks, although this definition can be extended to include arbitrary morphemes.

The nodes of the graph are called *units*. A unit may be either (i) a terminal or (ii) several elements jointly viewed as a single entity according to some semantic or cognitive consideration. In many cases, a non-terminal unit is comprised of a single relation and the units it applies to (its arguments), although in some cases it may also contain secondary relations. Hierarchy is formed by using units as arguments or relations in other units.

Categories are annotated over the graph’s edges,

and represent the descendant unit’s role in forming the semantics of the parent unit. Therefore, the internal structure of a unit is represented by its outbound edges and their categories, while the roles a unit plays in the relations it participates in are represented by its inbound edges.

We note that UCCA’s structures reflect a single interpretation of the text. Several discretely different interpretations (e.g., high vs. low PP attachments) may therefore yield several different UCCA annotations.

UCCA is a multi-layered formalism, where each layer specifies the relations it encodes. The question of which relations will be annotated (equivalently, which units will be formed) is determined by the layer in question. For example, consider “John kicked his ball”, and assume our current layer encodes the relations expressed by “kicked” and by “his”. In that case, the unit “his” has a single argument<sup>1</sup> (“ball”), while “kicked” has two (“John” and “his ball”). Therefore, the units of the sentence are the terminals (which are always units), “his ball” and “John kicked his ball”. The latter two are units by virtue of expressing a relation along with its arguments. See Figure 2(a) for a graph representation of this example.

For a brief comparison of the UCCA formalism with other dependency annotations see Section 5.

### 2.2 The UCCA Foundational Layer

The foundational layer is designed to cover the entire text, so that each word participates in at least one unit. It focuses on argument structures of verbal, nominal and adjectival predicates and the inter-relations between them. Argument structure phenomena are considered basic by many approaches to semantic and grammatical representation, and have a high applicative value, as demonstrated by their extensive use in NLP.

The foundational layer views the text as a collection of *Scenes*. A Scene can describe some movement or action, or a temporally persistent state. It generally has a temporal and a spatial dimension, which can be specific to a particular time and place, but can also describe a schematized event which refers to many events by highlighting a common meaning component. For example, the Scene “John loves bananas” is a schematized event, which refers to John’s disposition towards bananas without making any temporal or spatial

---

<sup>1</sup>The anaphoric aspects of “his” are not considered part of the current layer (see Section 2.3).

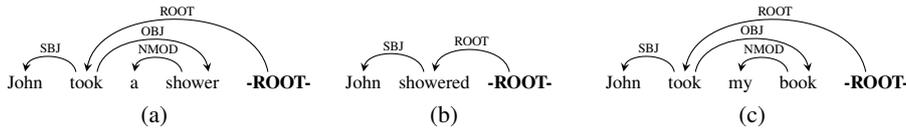


Figure 1: CoNLL-style dependency annotations. Note that (a) and (c), which have different semantics but superficially similar syntax, have the same annotation.

Abb.	Category	Short Definition
<b>Scene Elements</b>		
P	<b>Process</b>	The main relation of a Scene that evolves in time (usually an action or movement).
S	<b>State</b>	The main relation of a Scene that does not evolve in time.
A	<b>Participant</b>	A participant in a Scene in a broad sense (including locations, abstract entities and Scenes serving as arguments).
D	<b>Adverbial</b>	A secondary relation in a Scene (including temporal relations).
<b>Elements of Non-Scene Units</b>		
C	<b>Center</b>	Necessary for the conceptualization of the parent unit.
E	<b>Elaborator</b>	A non-Scene relation which applies to a single Center.
N	<b>Connector</b>	A non-Scene relation which applies to two or more Centers, highlighting a common feature.
R	<b>Relator</b>	All other types of non-Scene relations. Two varieties: (1) Rs that relate a C to some super-ordinate relation, and (2) Rs that relate two Cs pertaining to different aspects of the parent unit.
<b>Inter-Scene Relations</b>		
H	<b>Parallel Scene</b>	A Scene linked to other Scenes by regular linkage (e.g., temporal, logical, purposive).
L	<b>Linker</b>	A relation between two or more Hs (e.g., “when”, “if”, “in order to”).
G	<b>Ground</b>	A relation between the speech event and the uttered Scene (e.g., “surprisingly”, “in my opinion”).
<b>Other</b>		
F	<b>Function</b>	Does not introduce a relation or participant. Required by the structural pattern it appears in.

Table 1: The complete set of categories in UCCA’s foundational layer.

specifications. The definition of a Scene is motivated cross-linguistically and is similar to the semantic aspect of the definition of a “clause” in Basic Linguistic Theory<sup>2</sup>.

Table 1 provides a concise description of the categories used by the foundational layer<sup>3</sup>. We turn to a brief description of them.

**Simple Scenes.** Every Scene contains one main relation, which is the anchor of the Scene, the most important relation it describes (similar to frame-evoking lexical units in FrameNet (Baker et al., 1998)). We distinguish between static Scenes, that describe a temporally persistent state, and processual Scenes that describe a temporally evolving event, usually a movement or an action. The main relation receives the category *State* (*S*) in static and *Process* (*P*) in processual Scenes. We note that the S-P distinction is introduced here mostly for practical purposes, and that both categories can be viewed as sub-categories of the more abstract category Main Relation.

A Scene contains one or more *Participants* (*A*).

<sup>2</sup>As UCCA annotates categories on its edges, Scene nodes bear no special indication. They can be identified by examining the labels on their outgoing edges (see below).

<sup>3</sup>Repeated here with minor changes from (Abend and Rappoport, 2013), which focuses on the categories themselves.

This category subsumes concrete and abstract participants as well as embedded Scenes (see below). Scenes may also contain secondary relations, which are marked as *Adverbials* (*D*).

The above categories are indifferent to the syntactic category of the Scene-evoking unit, be it a verb, a noun, an adjective or a preposition. For instance, in the Scene “The book is in the garden”, “is in” is the *S*, while “the book” and “the garden” are *As*. In “Tomatoes are red”, the main static relation is “are red”, while “Tomatoes” is an *A*.

The foundational layer designates a separate set of categories to units that do not evoke a Scene. *Centers* (*C*) are the sub-units of a non-Scene unit that are necessary for the unit to be conceptualized and determine its semantic type. There can be one or more *Cs* in a non-Scene unit<sup>4</sup>.

Other sub-units of non-Scene units are categorized into three types. First, units that apply to a single *C* are annotated as *Elaborators* (*E*). For instance, “big” in “big dogs” is an *E*, while “dogs” is a *C*. We also mark determiners as *Es* in this coarse-grained layer<sup>5</sup>. Second, relations that relate two or

<sup>4</sup>By allowing several *Cs* we avoid the difficulties incurred by the common single head assumption. In some cases the *Cs* are inferred from context and can be implicit.

<sup>5</sup>Several *Es* that apply to a single *C* are often placed in

more Cs, highlighting a common feature or role (usually coordination), are called *Connectors (N)*. See an example in Figure 2(b).

*Relators (R)* cover all other types of relations between two or more Cs. Rs appear in two main varieties. In one, Rs relate a single entity to a super-ordinate relation. For instance, in “I heard noise in the kitchen”, “in” relates “the kitchen” to the Scene it is situated in. In the other, Rs relate two units pertaining to different aspects of the same entity. For instance, in “bottom of the sea”, “of” relates “bottom” and “the sea”, two units that refer to different aspects of the same entity.

Some units do not introduce a new relation or entity into the Scene, and are only part of the formal pattern in which they are situated. Such units are marked as *Functions (F)*. For example, in the sentence “it is customary for John to come late”, the “it” does not refer to any specific entity or relation and is therefore an F.

Two example annotations of simple Scenes are given in Figure 2(a) and Figure 2(b).

**More complex cases.** UCCA allows units to participate in more than one relation. This is a natural requirement given the wealth of distinctions UCCA is designed to accommodate. Already in the foundational layer of UCCA, the need arises for multiple parents. For instance, in “John asked Mary to join him”, “Mary” is a Participant of both the “asking” and the “joining” Scenes.

In some cases, an entity or relation is prominent in the interpretation of the Scene, but is not mentioned explicitly anywhere in the text. We mark such entities as *Implicit Units*. Implicit units are identical to terminals, except that they do not correspond to a stretch of text. For example, “playing games is fun” has an implicit A which corresponds to the people playing the game.

UCCA annotates inter-Scene relations (linkage) and, following Basic Linguistic Theory, distinguishes between three major types of linkage. First, a Scene can be an A in another Scene. For instance, in “John said he must leave”, “he must leave” is an A inside the Scene evoked by “said”. Second, a Scene may be an E of an entity in another Scene. For instance, in “the film we saw yesterday was wonderful”, “film we saw yesterday” is a Scene that serves as an E of “film”, which is both an A in the Scene and the Center of an A in the

a flat structure. In general, the coarse-grained foundational layer does not try to resolve fine scope issues.

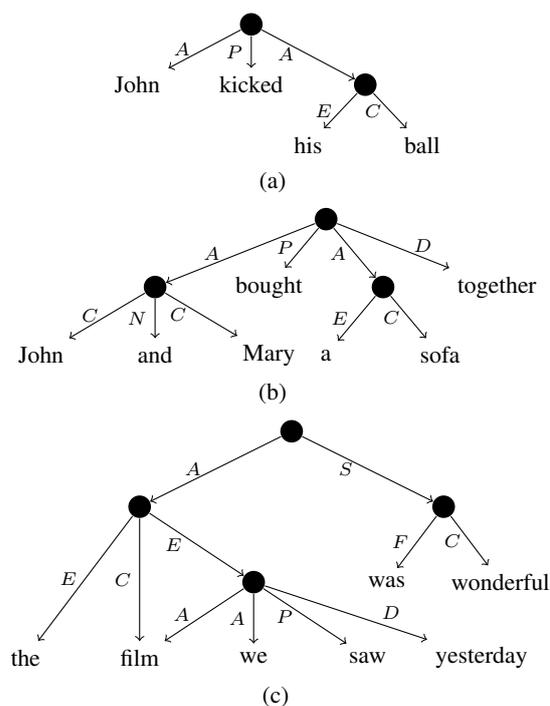


Figure 2: Examples of UCCA annotation graphs.

Scene evoked by “wonderful” (see Figure 2(c)).

A third type of linkage covers all other cases, e.g., temporal, causal and conditional inter-Scene relations. The linked Scenes in such cases are marked as *Parallel Scenes (H)*. The units specifying the relation between Hs are marked as *Linkers (L)*<sup>6</sup>. As with other relations in UCCA, Linkers and the Scenes they link are bound by a unit.

Unlike common practice in grammatical annotation, linkage relations in UCCA can cross sentence boundaries, as can relations represented in other layers (e.g., coreference). UCCA therefore annotates texts comprised of several paragraphs and not individual sentences (see Section 3).

**Example sentences.** Following are complete annotations of two abbreviated example sentences from our corpus (see Section 3).

“Golf became a passion for his oldest daughter: she took daily lessons and became very good, reaching the Connecticut Golf Championship.”

This sentence contains four Scenes, evoked by “became a passion”, “took daily lessons”, “became very good” and “reaching”. The individual Scenes are annotated as follows:

1. “Golf<sub>A</sub> [became<sub>E</sub> a<sub>E</sub> passion<sub>C</sub>]<sub>P</sub> [for<sub>R</sub> his<sub>E</sub> oldest<sub>E</sub> daughter<sub>C</sub>]<sub>A</sub>”

<sup>6</sup>It is equally plausible to include Linkers for the other two linkage types. This is not included in the current layer.

2. “she<sub>A</sub> [took<sub>F</sub> [daily<sub>E</sub> lessons<sub>C</sub>]<sub>C</sub>]<sub>P</sub>”
3. “she<sub>A</sub> ... [became<sub>E</sub> [very<sub>E</sub> good<sub>C</sub>]<sub>C</sub>]<sub>S</sub>”
4. “she<sub>A</sub> ... reaching<sub>P</sub> [the<sub>E</sub> Connecticut<sub>E</sub> Golf<sub>E</sub> Championship<sub>C</sub> ]<sub>A</sub>”

There is only one explicit Linker in this sentence (“and”), which links Scenes (2) and (3). None of the Scenes is an A or an E in the other, and they are therefore all marked as Parallel Scenes. We also note that in the case of the light verb construction “took lessons” and the copula clauses “became good” and “became a passion”, the verb is not the Center of the main relation, but rather the following noun or adjective. We also note that the unit “she” is an A in Scenes (2), (3) and (4).

We turn to our second example:

“Cukor encouraged the studio to  
accept her demands.”

This sentence contains three Scenes, evoked by “encouraged”, “accept” and “demands”:

1. Cukor<sub>A</sub> encouraged<sub>P</sub> [the<sub>E</sub> studio<sub>C</sub>]<sub>A</sub> [to<sub>R</sub> [accept her demands]<sub>C</sub> ]<sub>A</sub>
2. [the studio]<sub>A</sub> ... accept<sub>P</sub> [her demands]<sub>A</sub>
3. her<sub>A</sub> demands<sub>P</sub> **IMP**<sub>A</sub>

Scenes (2) and (3) act as Participants in Scenes (1) and (2) respectively. In Scene (2), there is an implicit Participant which corresponds to whatever was demanded. Note that “her demands” is a Scene, despite being a noun phrase.

### 2.3 UCCA’s Multi-layered Structure

Additional layers may refine existing relations or otherwise annotate a complementary set of distinctions. For instance, a refinement layer can categorize linkage relations according to their semantic types (e.g., temporal, purposive, causal) or provide tense distinctions for verbs. Another immediate extension to UCCA’s foundational layer can be the annotation of coreference relations. Recall the example “John kicked his ball”. A coreference layer would annotate a relation between “John” and “his” by introducing a new node whose descendants are these two units. The fact that this node represents a coreference relation would be represented by a label on the edge connecting them to the coreference node.

There are three common ways to extend an annotation graph. First, by adding a relation that relates previously established units. This is done by introducing a new node whose descendants are the related units. Second, by adding an intermediate

	Passage #					
	1	2	3	4	5	6
# Sents.	8	20	23	14	13	15
# Tokens	259	360	343	322	316	393
ITA	67.3	74.1	71.2	73.5	77.8	81.1
Vs. Gold	72.4	76.7	75.5	75.7	79.5	84.2
Correction	<b>93.7</b>					

Table 2: The upper part of the table presents the number of sentences and the number of tokens in the first passages used for the annotator training. The middle part presents the average F-scores obtained by the annotators throughout these passages. The first row presents the average F-score when comparing the annotations of the different annotators among themselves. The second row presents the average F-score when comparing them to a “gold standard”. The bottom row shows the average F-score between an annotated passage of a trained annotator and its manual correction by an expert. It is higher due to *conforming analyses* (see text). All F-scores are in percents.

unit between a parent unit and some of its sub-units. For instance, consider “he replied foolishly” and “he foolishly replied”. A layer focusing on Adverbial scope may refine the flat Scene structure assigned by the foundational layer, expressing the scope of “foolishly” over the relation “replied” in the first case, and over the entire Scene in the second. Third, by adding sub-units to a terminal. For instance, consider “gave up”, an expression which the foundational layer considers atomic. A layer that annotates tense can break the expression into “gave” and “up”, in order to annotate “gave” as the tense-bearing unit.

Although a more complete discussion of the formalism is beyond the scope of this paper, we note that the formalism is designed to allow different annotation layers to be defined and annotated independently of one another, in order to facilitate UCCA’s construction through a community effort.

### 3 A UCCA-Annotated Corpus

The annotated text is mostly based on English Wikipedia articles for celebrities. We have chosen this genre as it is an inclusive and diverse domain, which is still accessible to annotators from varied backgrounds.

For the annotation process, we designed and implemented a web application tailored for UCCA’s annotation. A sample of the corpus containing roughly 5K tokens, as well as the annotation application can be found in our website<sup>7</sup>.

UCCA’s annotations are not confined to a single sentence. The annotation is therefore carried out in passages of 300-400 tokens. After its an-

<sup>7</sup>[www.cs.huji.ac.il/~omria01](http://www.cs.huji.ac.il/~omria01)

notation, a passage is manually corrected before being inserted into the repository.

The section of the corpus annotated thus far contains 56890 tokens in 148 annotated passages (average length of 385 tokens). Each passage contains 450 units on average and 42.2 Scenes. Each Scene contains an average of 2 Participants and 0.3 Adverbials. 15% of the Scenes are static (contain an S as the main relation) and the rest are dynamic (containing a P). The average number of tokens in a Scene (excluding punctuation) is 10.7. 18.3% of the Scenes are Participants in another Scene, 11.4% are Elaborator Scenes and the remaining are Parallel Scenes. A passage contains an average of 11.2 Linkers.

**Inter-annotator agreement.** We employ 4 annotators with varying levels of background in linguistics. Two of the annotators have no background in linguistics, one took an introductory course and one holds a Bachelor's degree in linguistics. The training process of the annotators lasted 30–40 hours, which includes the time required for them to get acquainted with the web application. As this was the first large-scale trial with the UCCA scheme, some modifications to the scheme were made during the annotator's training. We therefore expect the training process to be even faster in later distributions.

There is no standard evaluation measure for comparing two grammatical annotations in the form of labeled DAGs. We therefore converted UCCA to constituency trees<sup>8</sup> and, following standard practice, computed the number of brackets in both trees that match in both span and label. We derive an F-score from these counts.

Table 2 presents the inter-annotator agreement in the training phase. The four annotators were given the same passage in each of these cases. In addition, a “gold standard” was annotated by the authors of this paper. The table presents the average F-score between the annotators, as well as the average F-score when comparing to the gold standard. Results show that although it represents complex hierarchical structures, the UCCA scheme is learned quickly and effectively.

We also examined the influence of prior linguistic background on the results. In the first passage there was a substantial advantage to the annotators

<sup>8</sup>In cases a unit had multiple parents, we discarded all but one of its incoming edges. This resulted in discarding 1.9% of the edges. We applied a simple normalization procedure to the resulting trees.

who had prior training in linguistics. The obtained F-scores when comparing to a gold standard, ordered decreasingly according to the annotator's acquaintance with linguistics, were 78%, 74.4%, 69.5% and 67.8%. However, this performance gap quickly vanished. Indeed, the obtained F-scores, again compared to a gold standard and averaged over the next five training passages, were (by the same order) 78.6%, 77.3%, 79.2% and 78%.

This is an advantage of UCCA over other syntactic annotation schemes that normally require highly proficient annotators. For instance, both the PTB and the Prague Dependency Treebank (Böhmová et al., 2003) employed annotators with extensive linguistic background. Similar findings to ours were reported in the PropBank project, which successfully employed annotators with various levels of linguistic background. We view this as a major advantage of semantic annotation schemes over their syntactic counterparts, especially given the huge amount of manual labor required for large syntactic annotation projects.

The UCCA interface allows for multiple non-contradictory (“conforming”) analyses of a stretch of text. It assumes that in some cases there is more than one acceptable option, each highlighting a different aspect of meaning of the analyzed utterance (see below). This makes the computation of inter-annotator agreement fairly difficult. It also suggests that the above evaluation is excessively strict, as it does not take into account such conforming analyses. To address this issue, we conducted another experiment where an expert annotator corrected the produced annotations. Comparing the corrected versions to the originals, we found that F-scores are typically in the range of 90%–95%. An average taken over a sample of passages annotated by all four annotators yielded an F-score of 93.7%.

It is difficult to compare the above results to the inter-annotator agreement of other projects for two reasons. First, many existing schemes are based on other annotation schemes or heavily rely on automatic tools for providing partial annotations. Second, some of the most prominent annotation projects do not provide reliable inter-annotator agreement scores (Artstein and Poesio, 2008).

A recent work that did report inter-annotator agreement in terms of bracketing F-score is an annotation project of the PTB's noun phrases with more elaborate syntactic structure (Vadas and Cur-

ran, 2011). They report an agreement of 88.3% in a scenario where their two annotators worked separately. Note that this task is much more limited in scope than UCCA (annotates noun phrases instead of complete passages in UCCA; uses 2 categories instead of 12 in UCCA). Nevertheless, the obtained inter-annotator agreement is comparable.

**Disagreement examples.** Here we discuss two major types of disagreements that recurred in the training process. The first is the distinction between Elaborators and Centers. In most cases this distinction is straightforward, particularly where one sub-unit determines the semantic type of the parent unit, while its siblings add more information to it (e.g., “truck<sub>E</sub> company<sub>C</sub>” is a type of a company and not of a truck). Some structures do not nicely fall into this pattern. One such case is with apposition. In the example “the Fox drama Glory days”, both “the Fox drama” and “Glory days” are reasonable candidates for being a Center, which results in disagreements.

Another case is the distinction between Scenes and non-Scene relations. Consider the example “[John’s portrayal of the character] has been described as ...”. The sentence obviously contains two scenes, one in which John portrays a character and another where someone describes John’s doings. Its internal structure is therefore “John’s<sub>A</sub> portrayal<sub>P</sub> [of the character]<sub>A</sub>”. However, the syntactic structure of this unit leads annotators at times into analyzing the subject as a non-Scene relation whose C is “portrayal”.

Static relations tend to be more ambiguous between a Scene and a non-Scene interpretation. Consider “Jane Smith (née Ross)”. It is not at all clear whether “née Ross” should be annotated as a Scene or not. Even if we do assume it is a Scene, it is not clear whether the Scene it evokes is her Scene of birth, which is dynamic, or a static Scene which can be paraphrased as “originally named Ross”. This leads to several conforming analyses, each expressing a somewhat different conceptualization of the Scene. This central notion will be more elaborately addressed in future work.

We note that all of these disagreements can be easily resolved by introducing an additional layer focusing on the construction in question.

#### 4 UCCA’s Benefits to Semantic Tasks

UCCA’s relative insensitivity to syntactic forms has potential benefits for a wide variety of seman-

tic tasks. This section briefly demonstrates these benefits through a number of examples.

Recall the example “John took a shower” (Section 1). UCCA annotates the sentence as a single Scene, with a single Participant and a processual main relation: “John<sub>A</sub> [took<sub>F</sub> [a<sub>E</sub> shower<sub>C</sub>]<sub>C</sub>]<sub>P</sub>”. The paraphrase “John showered” is annotated similarly: “John<sub>A</sub> showered<sub>P</sub>”. The structure is also preserved under translation to other languages, such as German (“John<sub>A</sub> duschte<sub>P</sub>”, where “duschte” is a verb), or Portuguese “John<sub>A</sub> [tomou<sub>F</sub> banho<sub>C</sub>]<sub>P</sub>” (literally, John took shower). In all of these cases, UCCA annotates the example as a Scene with an A and a P, whose Center is a word expressing the notion of showering.

Another example is the sentence “John does not have any money”. The foundational layer of UCCA annotates negation units as Ds, which yields the annotation “John<sub>A</sub> [does<sub>F</sub>]<sub>S</sub>- not<sub>D</sub> [have<sub>C</sub>]<sub>-S</sub> [any<sub>E</sub> money<sub>C</sub>]<sub>A</sub>” (where “does ... have” is a discontinuous unit)<sup>9</sup>. This sentence can be paraphrased as “John<sub>A</sub> has<sub>P</sub> no<sub>D</sub> money<sub>A</sub>”. UCCA reflects the similarity of these two sentences, as it annotates both cases as a single Scene which has two Participants and a negation. A syntactic scheme would normally annotate “no” in the second sentence as a modifier of “money”, and “not” as a negation of “have”.

The value of UCCA’s annotation can again be seen in translation to languages that have only one of these forms. For instance, the German translation of this sentence, “John<sub>A</sub> hat<sub>S</sub> kein<sub>D</sub> Geld<sub>A</sub>”, is a literal translation of “John has no money”. The Hebrew translation of this sentence is “eyn le john kesef” (literally, “there-is-no to John money”). The main relation here is therefore “eyn” (there-is-no) which will be annotated as *S*. This yields the annotation “eyn<sub>S</sub> [le<sub>R</sub> John<sub>C</sub>]<sub>A</sub> kesef<sub>A</sub>”.

The UCCA annotation in all of these cases is composed of two Participants and a State. In English and German, the negative polarity unit is represented as a D. The negative polarity of the Hebrew “eyn” is represented in a more detailed layer.

As a third example, consider the two sentences “There are children playing in the park” and “Children are playing in the park”. The two sentences have a similar meaning but substantially different syntactic structures. The first contains two clauses, an existential main clause (headed by “there are”)

<sup>9</sup>The foundational layer places “not” in the Scene level to avoid resolving fine scope issues (see Section 2)

and a subordinate clause (“playing in the park”). The second contains a simple clause headed by “playing”. While the parse trees of these sentences are very different, their UCCA annotation in the foundational layer differ only in terms of Function units: “Children<sub>A</sub> [are<sub>F</sub> playing<sub>C</sub>]<sub>P</sub> [in<sub>R</sub> the<sub>E</sub> park<sub>C</sub>]<sub>A</sub>” and “There<sub>F</sub> are<sub>F</sub> children<sub>A</sub> [playing]<sub>P</sub> [in<sub>R</sub> the<sub>E</sub> park<sub>C</sub>]<sub>A</sub>”<sup>10</sup>.

Aside from machine translation, a great variety of semantic tasks can benefit from a scheme that is relatively insensitive to syntactic variation. Examples include text simplification (e.g., for second language teaching) (Siddharthan, 2006), paraphrase detection (Dolan et al., 2004), summarization (Knight and Marcu, 2000), and question answering (Wang et al., 2007).

## 5 Related Work

In this section we compare UCCA to some of the major approaches to grammatical representation in NLP. We focus on English, which is the most studied language and the focus of this paper.

Syntactic annotation schemes come in many forms, from lexical categories such as POS tags to intricate hierarchical structures. Some formalisms focus particularly on syntactic distinctions, while others model the syntax-semantics interface as well (Kaplan and Bresnan, 1981; Pollard and Sag, 1994; Joshi and Schabes, 1997; Steedman, 2001; Sag, 2010, *inter alia*). UCCA diverges from these approaches in aiming to abstract away from specific syntactic forms and to only represent semantic distinctions. Put differently, UCCA advocates an approach that treats syntax as a hidden layer when learning the mapping between form and meaning, while existing syntactic approaches aim to model it manually and explicitly.

UCCA does not build on any other annotation layers and therefore implicitly assumes that semantic annotation can be learned directly. Recent work suggests that indeed structured prediction methods have reached sufficient maturity to allow direct learning of semantic distinctions. Examples include Naradowsky et al. (2012) for semantic role labeling and Kwiatkowski et al. (2010) for semantic parsing to logical forms. While structured prediction for the task of predicting tree structures is already well established (e.g., (Suzuki et al.,

2009)), recent work has also successfully tackled the task of predicting semantic structures in the form of DAGs (Jones et al., 2012).

The most prominent annotation scheme in NLP for English syntax is the Penn Treebank. Many syntactic schemes are built or derived from it. An increasingly popular alternative to the PTB are dependency structures, which are usually represented as trees whose nodes are the words of the sentence (Ivanova et al., 2012). Such representations are limited due to their inability to naturally represent constructions that have more than one head, or in which the identity of the head is not clear. They also face difficulties in representing units that participate in multiple relations. UCCA proposes a different formalism that addresses these problems by introducing a new node for every relation (cf. (Sangati and Mazza, 2009)).

Several annotated corpora offer a joint syntactic and semantic representation. Examples include the Groningen Meaning bank (Basile et al., 2012), Treebank Semantics (Butler and Yoshimoto, 2012) and the Lingo Redwoods treebank (Oepen et al., 2004). UCCA diverges from these projects in aiming to abstract away from syntactic variation, and is therefore less coupled with a specific syntactic theory.

A different strand of work addresses the construction of an interlingual representation, often with a motivation of applying it to machine translation. Examples include the UNL project (Uchida and Zhu, 2001), the IAMTC project (Dorr et al., 2010) and the AMR project (Banarescu et al., 2012). These projects share with UCCA their emphasis on cross-linguistically valid annotations, but diverge from UCCA in three important respects. First, UCCA emphasizes the notion of a multi-layer structure where the basic layers are maximally coarse-grained, in contrast to the above works that use far more elaborate representations. Second, from a theoretical point of view, UCCA differs from these works in aiming to represent conceptual semantics, building on works in Cognitive Linguistics (e.g., (Langacker, 2008)). Third, unlike interlingua that generally define abstract representations that may correspond to several different texts, UCCA incorporates the text into its structure, thereby facilitating learning.

Semantic role labeling (SRL) schemes bear similarity to the foundational layer, due to their focus on argument structure. The leading SRL ap-

<sup>10</sup>The two sentences are somewhat different in terms of their information structure (Van Valin Jr., 2005), which is represented in a more detailed UCCA layer.

proaches are PropBank (Palmer et al., 2005) and NomBank (Meyers et al., 2004) on the one hand, and FrameNet (Baker et al., 1998) on the other. At this point, all these schemes provide a more fine-grained set of categories than UCCA.

PropBank and NomBank are built on top of the PTB annotation, and provide for each verb (PropBank) and noun (NomBank), a delineation of their arguments and their categorization into semantic roles. Their structures therefore follow the syntax of English quite closely. UCCA is generally less tailored to the syntax of English (e.g., see secondary verbs (Dixon, 2005)).

Furthermore, PropBank and NomBank do not annotate the internal structure of their arguments. Indeed, the construction of the commonly used semantic dependencies derived from these schemes (Surdeanu et al., 2008) required a set of syntactic head percolation rules to be used. These rules are somewhat arbitrary (Schwartz et al., 2011), do not support multiple heads, and often reflect syntactic rather than semantic considerations (e.g., “millions” is the head of “millions of dollars”, while “dollars” is the head of “five million dollars”).

Another difference is that PropBank and NomBank each annotate only a subset of predicates, while UCCA is more inclusive. This difference is most apparent in cases where a single complex predicate contains both nominal and verbal components (e.g., “limit access”, “take a shower”). In addition, neither PropBank nor NomBank address copula clauses, despite their frequency. Finally, unlike PropBank and NomBank, UCCA’s foundational layer annotates linkage relations.

In order to quantify the similarity between UCCA and PropBank, we annotated 30 sentences from the PropBank corpus with their UCCA annotations and converted the outcome to PropBank-style annotations<sup>11</sup>. We obtained an unlabeled F-score of 89.4% when comparing to PropBank, which indicates that PropBank-style annotations are generally derivable from UCCA’s. The disagreement between the schemes reflects both annotation conventions and principle differences, some of which were discussed above.

The FrameNet project (Baker et al., 1998)

<sup>11</sup>The experiment was conducted on the first 30 sentences of section 02. The identity of the predicates was determined according to the PropBank annotation. We applied a simple conversion procedure that uses half a dozen rules that are not conditioned on any lexical item. We used a strict evaluation that requires an exact match in the argument’s boundaries.

proposes a comprehensive approach to semantic roles. It defines a lexical database of Frames, each containing a set of possible frame elements and their semantic roles. It bears similarity to UCCA both in its use of Frames, which are a context-independent abstraction of UCCA’s Scenes, and in its emphasis on semantic rather than distributional considerations. However, despite these similarities, FrameNet focuses on constructing a lexical resource covering specific cases of interest, and does not provide a fully annotated corpus of naturally occurring text. UCCA’s foundational layer can be seen as a complementary effort to FrameNet, as it focuses on high-coverage, coarse-grained annotation, while FrameNet is more fine-grained at the expense of coverage.

## 6 Conclusion

This paper presented Universal Conceptual Cognitive Annotation (UCCA), a novel framework for semantic representation. We described the foundational layer of UCCA and the compilation of a UCCA-annotated corpus. We demonstrated UCCA’s relative insensitivity to paraphrases and cross-linguistic syntactic variation. We also discussed UCCA’s accessibility to annotators with no background in linguistics, which can alleviate the almost prohibitive annotation costs of large syntactic annotation projects.

UCCA’s representation is guided by conceptual notions and has its roots in the Cognitive Linguistics tradition and specifically in Cognitive Grammar (Langacker, 2008). These theories represent the meaning of an utterance according to the mental representations it evokes and not according to its reference in the world. Future work will explore options to further reduce manual annotation, possibly by combining texts with visual inputs during training.

We are currently attempting to construct a parser for UCCA and to apply it to several semantic tasks, notably English-French machine translation. Future work will also discuss UCCA’s portability across domains. We intend to show that UCCA, which is less sensitive to the idiosyncrasies of a specific domain, can be easily adapted to highly dynamic domains such as social media.

**Acknowledgements.** We would like to thank Tomer Eshet for partnering in the development of the web application and to Amit Beka for his help with UCCA’s software and development set.

## References

- Omri Abend and Ari Rappoport. 2013. UCCA: A semantics-based grammatical annotation scheme. In *IWCS '13*, pages 1–12.
- Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.
- Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley Framenet project. In *ACL-COLING '98*, pages 86–90.
- Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2012. Abstract meaning representation (AMR) 1.0 specification. <http://www.isi.edu/natural-language/people/amr-guidelines-10-31-12.pdf>.
- Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *LREC '12*, pages 3196–3200.
- Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2003. The Prague Dependency Treebank. *Treebanks*, pages 103–127.
- Alistair Butler and Kei Yoshimoto. 2012. Banking meaning representations from treebanks. *Linguistic Issues in Language Technology*, 7(1).
- Alexander Clark and Shalom Lappin. 2010. *Linguistic Nativism and the Poverty of the Stimulus*. Wiley-Blackwell.
- Robert M. W. Dixon. 2005. *A Semantic Approach To English Grammar*. Oxford University Press.
- Robert M. W. Dixon. 2010a. *Basic Linguistic Theory: Methodology*, volume 1. Oxford University Press.
- Robert M. W. Dixon. 2010b. *Basic Linguistic Theory: Grammatical Topics*, volume 2. Oxford University Press.
- Robert M. W. Dixon. 2012. *Basic Linguistic Theory: Further Grammatical Topics*, volume 3. Oxford University Press.
- Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING '04*, pages 350–356.
- Bonnie Dorr, Rebecca Passonneau, David Farwell, Rebecca Green, Nizar Habash, Stephen Helmreich, Edward Hovy, Lori Levin, Keith Miller, Teruko Mitamura, Owen Rambow, and Advaith Siddharthan. 2010. Interlingual annotation of parallel text corpora: A new framework for annotation and evaluation. *Natural Language Engineering*, 16(3):197–243.
- Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012. Who did what to whom?: A contrastive study of syntacto-semantic dependencies. In *LAW '12*, pages 2–11.
- Bevan Jones, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, and Kevin Knight. 2012. Semantics-based machine translation with hyperedge replacement grammars. In *COLING '12*, pages 1359–1376.
- Aravind K. Joshi and Yves Schabes. 1997. Tree-adjointing grammars. *Handbook Of Formal Languages*, 3:69–123.
- Ronald M. Kaplan and Joan Bresnan. 1981. *Lexical-Functional Grammar: A Formal System For Grammatical Representation*. Massachusetts Institute Of Technology, Center For Cognitive Science.
- Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization – step one: Sentence compression. In *AAAI '00*, pages 703–710.
- Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *EMNLP '10*, pages 1223–1233.
- R.W. Langacker. 2008. *Cognitive Grammar: A Basic Introduction*. Oxford University Press, USA.
- Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. Annotating noun argument structure for Nombank. In *LREC '04*, pages 803–806.
- Jason Naradowsky, Sebastian Riedel, and David Smith. 2012. Improving NLP through marginalization of hidden syntactic structure. In *EMNLP '12*, pages 810–820.
- Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christopher D Manning. 2004. Lingo redwoods. *Research on Language and Computation*, 2(4):575–596.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):145–159.
- Carl Pollard and Ivan A. Sag. 1994. *Head-driven Phrase Structure Grammar*. University Of Chicago Press.
- Ivan A Sag. 2010. Sign-based construction grammar: An informal synopsis. *Sign-based Construction Grammar: CSLI Publications, Stanford*, pages 39–170.

- Federico Sangati and Chiara Mazza. 2009. An English dependency treebank à la Tesnière. In *TLT '09*, pages 173–184.
- Roy Schwartz, Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *ACL-HLT '11*, pages 663–672.
- Advait Siddharthan. 2006. Syntactic simplification and text cohesion. *Research on Language & Computation*, 4(1):77–109.
- Mark Steedman. 2001. *The Syntactic Process*. MIT Press.
- Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL '08*, pages 159–177.
- Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *EMNLP '09*, pages 551–560.
- Hiroshi Uchida and Meiyang Zhu. 2001. The universal networking language beyond machine translation. In *International Symposium on Language in Cyberspace*, pages 26–27.
- David Vadas and James R Curran. 2011. Parsing noun phrases in the Penn Treebank. *Computational Linguistics*, 37(4):753–809.
- Robert D. Van Valin Jr. 2005. *Exploring The Syntax-semantics Interface*. Cambridge University Press.
- Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *EMNLP-CoNLL '07*, pages 22–32.

# Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media

Weiwei Guo\* and Hao Li† and Heng Ji† and Mona Diab‡

\*Department of Computer Science, Columbia University

†Computer Science Department and Linguistic Department,  
Queens College and Graduate Center, City University of New York

‡Department of Computer Science, George Washington University

weiwei@cs.columbia.edu, {haoli.qc,hengjicuny}@gmail.com, mtdiab@gwu.edu

## Abstract

Many current Natural Language Processing [NLP] techniques work well assuming a large context of text as input data. However they become ineffective when applied to short texts such as Twitter feeds. To overcome the issue, we want to find a related newswire document to a given tweet to provide contextual support for NLP tasks. This requires robust modeling and understanding of the semantics of short texts.

The contribution of the paper is two-fold: 1. we introduce the Linking-Tweets-to-News task as well as a dataset of linked tweet-news pairs, which can benefit many NLP applications; 2. in contrast to previous research which focuses on lexical features within the short texts (text-to-word information), we propose a graph based latent variable model that models the inter short text correlations (text-to-text information). This is motivated by the observation that a tweet usually only covers one aspect of an event. We show that using tweet specific feature (hashtag) and news specific feature (named entities) as well as temporal constraints, we are able to extract text-to-text correlations, and thus completes the semantic picture of a short text. Our experiments show significant improvement of our new model over baselines with three evaluation metrics in the new task.

## 1 Introduction

Recently there has been an increasing interest in language understanding of Twitter messages. Researchers (Speriosui et al., 2011; Brody and Diakopoulos, 2011; Jiang et al., 2011) were in-

terested in sentiment analysis on Twitter feeds, and opinion mining towards political issues or politicians (Tumasjan et al., 2010; Conover et al., 2011). Others (Ramage et al., 2010; Jin et al., 2011) summarized tweets using topic models. Although these NLP techniques are mature, their performance on tweets inevitably degrades, due to the inherent sparsity in short texts. In the case of sentiment analysis, while people are able to achieve 87.5% accuracy (Maas et al., 2011) on a movie review dataset (Pang and Lee, 2004), the performance drops to 75% (Li et al., 2012) on a sentence level movie review dataset (Pang and Lee, 2005). The problem worsens when some existing NLP systems cannot produce any results given the short texts. Considering the following tweet:

*Pray for Mali...*

A typical event extraction/discovery system (Ji and Grishman, 2008) fails to discover the *war* event due to the lack of context information (Benson et al., 2011), and thus fails to shed light on the users focus/interests.

To enable the NLP tools to better understand Twitter feeds, we propose the task of linking a tweet to a news article that is relevant to the tweet, thereby augmenting the context of the tweet. For example, we want to supplement the implicit context of the above tweet with a news article such as the following entitled:

*State of emergency declared in Mali*

where abundant evidence can be fed into an off-the-shelf event extraction/discovery system. To create a gold standard dataset, we download tweets spanning over 18 days, each with a url linking to a news article of CNN or NYTIMES, as well as all the news of CNN and NYTIMES published during the period. The goal is to predict the url referred news article based on the text in each tweet.<sup>1</sup> We

<sup>1</sup>The data and code is publicly available at [www.cs.columbia.edu/~weiwei/](http://www.cs.columbia.edu/~weiwei/)

believe many NLP tasks will benefit from this task. In fact, in the topic modeling research, previous work (Jin et al., 2011) already showed that by incorporating webpages whose urls are contained in tweets, the tweet clustering purity score was boosted from 0.280 to 0.392.

Given the few number of words in a tweet (14 words on average in our dataset), the traditional high dimensional surface word matching is lossy and fails to pinpoint the news article. This constitutes a classic short text semantics impediment (Agirre et al., 2012). Latent variable models are powerful by going beyond the surface word level and mapping short texts into a low dimensional dense vector (Socher et al., 2011; Guo and Diab, 2012b). Accordingly, we apply a latent variable model, namely, the Weighted Textual Matrix Factorization [WTMF] (Guo and Diab, 2012b; Guo and Diab, 2012c) to both the tweets and the news articles. WTMF is a state-of-the-art unsupervised model that was tested on two short text similarity datasets: (Li et al., 2006) and (Agirre et al., 2012), which outperforms Latent Semantic Analysis [LSA] (Landauer et al., 1998) and Latent Dirichlet Allocation [LDA] (Blei et al., 2003) by a large margin. We employ it as a strong baseline in this task as it exploits and effectively models the missing words in a tweet, in practice adding thousands of more features for the tweet, by contrast LDA, for example, only leverages observed words (14 features) to infer the latent vector for a tweet.

Apart from the data sparseness, our dataset proposes another challenge: a tweet usually covers only one aspect of an event. In our previous example, the tweet only contains the location *Mali* while the event is about French army participated in Mali war. In this scenario, we would like to find the missing elements of the tweet such as *French, war* from other short texts, to complete the semantic picture of *Pray in Mali* tweet. One drawback of WTMF for our purposes is that it simply models the text-to-word information without leveraging the correlation between short texts. While this is acceptable on standard short text similarity datasets (data points are independently generated), it ignores some valuable information characteristically present in our dataset: (1) The tweet specific features such as hashtags. Hashtags prove to be a direct indication of the semantics of tweets (Ramage et al., 2010); (2) The news specific features

such as named entities in a document. Named entities acquired from a news document, typically with high accuracy using Named Entity Recognition [NER] tools, may be particularly informative. If two texts mention the same entities then they might describe the same event; (3) The temporal information in both genres (tweets and news articles). We note that there is a higher chance of event description overlap between two texts if their time of publication is similar.

In this paper, we study the problem of mining and exploiting correlations between texts using these features. Two texts may be considered related or complementary if they share a hashtag/NE or satisfies the temporal constraints. Our proposed latent variable model not only models text-to-word information, but also is aware of the text-to-text information (illustrated in Figure 1): two linked texts should have similar latent vectors, accordingly the semantic picture of a tweet is completed by receiving semantics from its related tweets. We incorporate this additional information in the WTMF model. We also show the different impact of the text-to-text relations in the tweet genre and news genre. We are able to achieve significantly better results than with a text-to-words WTMF model. This work can be regarded as a short text modeling approach that extends previous work however with a focus on combining the mining of information within short texts coupled with utilizing extra shared information across the short texts.

## 2 Task and Data

The task is given the text in a tweet, a system aims to find the most relevant news article. For gold standard data, we harvest all the tweets that have a single url link to a CNN or NYTIMES news article, dated from the 11th of Jan to the 27th of Jan, 2013. In evaluation, we consider this url-referred news article as the gold standard – the most relevant document for the tweet, and remove the url from the text of the tweet. We also collect all the news articles from both CNN and NYTIMES from RSS feeds during the same timeframe. Each tweet entry has the **published time, author, text**; each news entry contains **published time, title, news summary, url**. The tweet/news pairs are extracted by matching urls. We manually filtered “trivial” tweets where the tweet content is simply the news title or news summary. The final dataset results in

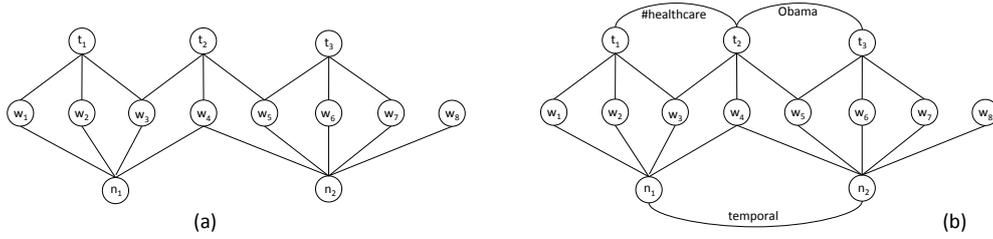


Figure 1: (a) WTMF. (b) WTMF-G: the tweet nodes  $t$  and news nodes  $n$  are connected by hashtags, named entities or temporal edges (for simplicity, the missing tokens are not shown in the figure)

34,888 tweets and 12,704 news articles.

It is worth noting that the news corpus is not restricted to current events. It covers various genres and topics, such as travel guides. e.g. *World’s most beautiful lakes*, and health issues, e.g. *The importance of a ‘stop day’*, etc.

### 2.1 Evaluation metric

For our task evaluation, ideally, we would like the system to be able to identify the news article specifically referred to by the url within each tweet in the gold standard. However, this is very difficult given the large number of potential candidates, especially those with slight variations. Therefore, following the Concept Definition Retrieval task in (Guo and Diab, 2012b) and (Steck, 2010) we use a metric for evaluating the ranking of the correct news article to evaluate the systems, namely,  $ATOP_t$ , *area under the TOPK<sub>t</sub>(k) recall curve for a tweet t*. Basically, it is the normalized ranking  $\in [0, 1]$  of the correct news article among all candidate news articles:  $ATOP_t = 1$  means the url-referred news article has the highest similarity value with the tweet (a correct NARU);  $ATOP_t = 0.95$  means the similarity value with correct news article is larger than 95% of the candidates, i.e. within the top 5% of the candidates.  $ATOP_t$  is calculated as follows:

$$ATOP_t = \int_0^1 TOPK_t(k) dk \quad (1)$$

where  $TOPK_t(k) = 1$  if the url referred news article is in the “top  $k$ ” list, otherwise  $TOPK_t(k) = 0$ . Here  $k \in [0, 1]$  is the relative position (when  $k = 1$ , it means all the candidates).

We also include other metrics to examine if the system is able to rank the url referred news article in the first few returned results: **TOP10** recall hit rate to evaluate whether the correct news is in the top 10 results, and **RR**, Reciprocal Rank=  $1/r$

(i.e.,  $RR = 1/3$  when the correct news article is ranked at the 3rd highest place).

## 3 Weighted Textual Matrix Factorization

The WTMF model (Guo and Diab, 2012a) has been successfully applied to the short text similarity task, achieving state-of-the-art unsupervised performance. This can be attributed to the fact that it models the missing tokens as features, thereby adding many more features for a short text. The missing words of a sentence are defined as all the vocabulary of the training corpus minus the observed words in a sentence. Missing words serve as negative examples for the semantics of a short text: the short text should not be related to the missing words.

As per (Guo and Diab, 2012b), the corpus is represented in a matrix  $X$ , where each cell stores the TF-IDF values of words. The rows of  $X$  are words and columns are short texts. As in Figure 2, matrix  $X$  is approximated by the product of a  $K \times M$  matrix  $P$  and a  $K \times N$  matrix  $Q$ . Accordingly, each sentence  $s_j$  is represented by a  $K$  dimensional latent vector  $Q_{\cdot,j}$ . Similarly a word  $w_i$  is generalized by  $P_{\cdot,i}$ . Therefore, the inner product of a word vector  $P_{\cdot,i}$  and a short text vector  $Q_{\cdot,j}$  is to approximate the cell  $X_{ij}$  (shaded part in Figure 2). In this way, the missing words are modeled by requiring the inner product of a word vector and short text vector to be close to 0 (the word and the short text should be irrelevant).

Since 99% cells in  $X$  are missing tokens (0 value), the impact of observed words is significantly diminished. Therefore a small weight  $w_m$  is assigned for each 0 cell (missing tokens) in the matrix  $X$  in order to preserve the influence of observed words.  $P$  and  $Q$  are optimized by minimize the objective function:

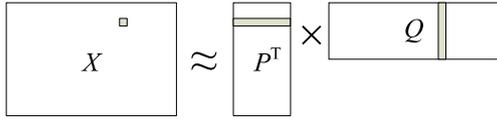


Figure 2: Weighted Textual Matrix Factorization

$$\sum_i \sum_j W_{ij} (P_{\cdot,i} \cdot Q_{\cdot,j} - X_{ij})^2 + \lambda \|P\|_2^2 + \lambda \|Q\|_2^2$$

$$W_{i,j} = \begin{cases} 1, & \text{if } X_{ij} \neq 0 \\ w_m, & \text{if } X_{ij} = 0 \end{cases} \quad (2)$$

where  $\lambda$  is a regularization term.

## 4 Creating Text-to-text Relations via Twitter/News Features

WTMF exploits the text-to-word information in a very nuanced way, while the dependency between texts is ignored. In this Section, we introduce how to create text-to-text relations.

### 4.1 Hashtags and Named Entities

Hashtags highlight the topics in tweets, e.g., *The #flu season has started*. We believe two tweets sharing the same hashtag should be related, hence we place a link between them to explicitly inform the model that these two tweets should be similar.

We find only 8,701 tweets out of 34,888 include hashtags. In fact, we observe many hashtag words are mentioned in tweets without explicitly being tagged with #. To overcome the hashtag sparseness issue, one can resort to keywords recommendation algorithms to mine hashtags for the tweets (Yang et al., 2012). In this paper, we adopt a simple but effective approach: we collect all the hashtags in the dataset, and automatically hashtag any word in a tweet if that word appears hashtagged in any other tweets. This process resulted in 33,242 tweets automatically labeled with hashtags. For each tweet, and for each hashtag it contains, we extract  $k$  tweets that contain this hashtag, assuming they are complementary to the target tweet, and link the  $k$  tweets to the target tweet. If there are more than  $k$  tweets found, we choose the top  $k$  ones that are most chronologically close to the target tweet. The statistics of links can be found in table 2.

Named entities are some of the most salient features in a news article. Directly applying Named Entity Recognition (NER) tools on news titles or

tweets results in many errors (Liu et al., 2011) due to the noise in the data, such as slang and capitalization. Accordingly, we first apply the NER tool on news summaries, then label named entities in the tweets in the same way as labeling the hashtags: if there is a string in the tweet that matches a named entity from the summaries, then it is labeled as a named entity in the tweet. 25,132 tweets are assigned at least one named entity.<sup>2</sup> To create the similar tweet set, we find  $k$  tweets that also contain the named entity.

### 4.2 Temporal Relations

Tweets published in the same time interval have a larger chance of being similar than those are not chronologically close (Wang and McCallum, 2006). However, we cannot simply assume any two tweets are similar only based on the timestamp. Therefore, for a tweet we link it to the  $k$  most similar tweets whose published time is within 24 hours of the target tweet’s timestamp. We use the similarity score returned by WTMF model to measure the similarity of two tweets.

We experimented with other features such as authorship. We note that it was not a helpful feature. While authorship information helps in the task of news/tweets recommendation for a *user* (Corso et al., 2005; Yan et al., 2012), the authorship information is too general for this task where we target on “recommending” a news article for a *tweet*.

### 4.3 Creating Relations on News

We extract the 3 subgraphs (based on hashtags, named entities and temporal) on news articles. However, automatically tagging hashtags or named entities leads to much worse performance (around 93% ATOP values, a 3% decrease from baseline WTMF). There are several reasons for this: 1. When a hashtag-matched word appears in a tweet, it is often related to the central meaning of the tweet, however news articles are generally much longer than tweets, resulting in many more hashtags/named entities matches even though these named entities may not be closely related. 2. The noise introduced during automatic NER accumulates much faster given the large number of named entities in news data. Therefore we only extract temporal relations for news articles.

<sup>2</sup>Note that there are some false positive named entities detected such as *apple*. We plan to address removing noisy named entities and hashtags in future work

## 5 WTMF on Graphs

We propose a novel model to incorporate the links generated as described in the previous section.

If two texts are connected by a link, it means they should be semantically similar. In the WTMF model, we would like the latent vectors of two text nodes  $Q_{\cdot,j_1}, Q_{\cdot,j_2}$  to be as similar as possible, namely that their cosine similarity to be close to 1. To implement this, we add a regularization term in the objective function of WTMF (equation 2) for each linked pairs  $Q_{\cdot,j_1}, Q_{\cdot,j_2}$ :

$$\delta \cdot \left( \frac{Q_{\cdot,j_1} \cdot Q_{\cdot,j_2}}{|Q_{\cdot,j_1}| |Q_{\cdot,j_2}|} - 1 \right)^2 \quad (3)$$

where  $|Q_{\cdot,j}|$  denotes the length of vector  $Q_{\cdot,j}$ . The coefficient  $\delta$  denotes the importance of the text-to-text links. A larger  $\delta$  means we put more weights on the text-to-text links and less on the text-to-word links. We refer to this model as WTMF-G (WTMF on graphs).

### 5.1 Inference

Alternating Least Square [ALS] is used for inference in weighted matrix factorization (Srebro and Jaakkola, 2003). However, ALS is no longer applicable here with the new regularization term (equation 3) involving the length of text vectors  $|Q_{\cdot,j}|$ , which is not in quadratic form. Therefore we approximate the objective function by treating the vector length  $|Q_{\cdot,j}|$  as fixed values during the ALS iterations:

$$\begin{aligned} P_{\cdot,i} &= (Q\tilde{W}^{(i)}Q^\top + \lambda I)^{-1} Q\tilde{W}^{(i)}X_{\cdot,i} \\ Q_{\cdot,j} &= (P\tilde{W}^{(j)}P^\top + \lambda I + \delta L_{(j)}^2 Q_{\cdot,s(j)} \text{diag}(L_{(s(j))}^2) Q_{\cdot,s(j)}^\top)^{-1} \\ &\quad (P\tilde{W}^{(j)}X_{j\cdot}^\top + \delta L_{(j)} Q_{\cdot,s(j)} L_{n(j)}) \end{aligned} \quad (4)$$

We define  $n(j)$  as the linked neighbors of short text  $j$ , and  $Q_{\cdot,n(j)}$  as the set of latent vectors of  $j$ 's neighbors. The reciprocal of length of these vectors in the current iteration are stored in  $L_{s(j)}$ . Similarly, the reciprocal of the length of the short text vector  $Q_{\cdot,j}$  is  $L_j$ .  $\tilde{W}^{(i)} = \text{diag}(W_{\cdot,i})$  is an  $M \times M$  diagonal matrix containing the  $i$ th row of weight matrix  $W$ . Due to limited space, the details of the optimization are not shown in this paper; they can be found in (Steck, 2010).

## 6 Experiments

### 6.1 Experiment Setting

**Corpora:** We use the same corpora as in (Guo and Diab, 2012b): Brown corpus (each sentence is

treated as a document), sense definitions of Wiktionary and Wordnet (Fellbaum, 1998). The tweets and news articles are also included in the corpus, generating 441,258 short texts and 5,149,122 words. The data is tokenized, POS-tagged by Stanford POS tagger (Toutanova et al., 2003), and lemmatized by WordNet::QueryData.pm. The value of each word in matrix  $X$  is its TF-IDF value in the short text.

**Baselines:** We present 4 baselines: 1. Information Retrieval model [IR], which simply treats a tweet as a document, and performs traditional surface word matching. 2. LDA- $\theta$  with Gibbs Sampling as inference method. We use the inferred topic distribution  $\theta$  as a latent vector to represent the tweet/news. 3. LDA-*wvec*. The problem with LDA- $\theta$  is the inferred topic distribution latent vector is very sparse with only a few non-zero values, resulting in many tweet/news pairs receiving a high similarity value as long as they are in the same topic domain. Hence following (Guo and Diab, 2012b), we first compute the latent vector of a word by  $P(z|w)$  (topic distribution per word), then average the word latent vectors weighted by TF-IDF values to represent the short text, which yields much better results. 4. WTMF. In these baselines, hashtags and named entities are simply treated as words.

To curtail variation in results due to randomness, each reported number is the average of 10 runs. For WTMF and WTMF-G, we assign the same initial random values and run 20 iterations. In both systems we fix the missing words weight as  $w_m = 0.01$  and regularization coefficient at  $\lambda = 20$ , which is the best condition of WTMF found in (Guo and Diab, 2012b; Guo and Diab, 2012c). For LDA- $\theta$  and LDA-*wvec*, we run Gibbs Sampling based LDA for 2000 iterations and average the model over the last 10 iterations.

**Evaluation:** The similarity between a tweet and a news article is measured by cosine similarity. A news article is represented as the concatenation of its title and its summary, which yields better performance.<sup>3</sup>

As in (Guo and Diab, 2012b), for each tweet, we collect the 1,000 news articles published prior to the tweet whose dates of publication are closest to that of the tweet.<sup>4</sup> The cosine similarity

<sup>3</sup>While these are separated, WTMF receive ATOP 95.558% for representing news article as titles and 94.385% for representing news article as summaries

<sup>4</sup>Ideally we want to include all the news articles published

Models	Parameters	ATOP		TOP10		RR	
		dev	test	dev	test	dev	test
IR	-	90.795%	90.743%	73.478%	74.103%	46.024%	46.281%
LDA- $\theta$	$\alpha = 0.05, \beta = 0.05$	81.368%	81.251%	32.328%	31.207%	13.134%	12.469%
LDA- <i>wvec</i>	$\alpha = 0.05, \beta = 0.05$	94.148%	94.196%	53.500%	53.952%	28.743%	27.904%
WTMF	-	95.964%	96.092%	75.327%	76.411%	45.310%	46.270%
WTMF-G	$k = 3, \delta = 3$	96.450%	96.543%	76.485%	77.479%	47.516%	48.665%
WTMF-G	$k = 5, \delta = 3$	<b>96.613%</b>	<b>96.701%</b>	76.029%	77.176%	47.197%	48.189%
WTMF-G	$k = 4, \delta = 3$	96.510%	96.610%	<b>77.782%</b>	<b>77.782%</b>	<b>47.917%</b>	<b>48.997%</b>

Table 1: ATOP Performance (latent dimension  $D = 100$  for LDA/WTMF/WTMF-G)

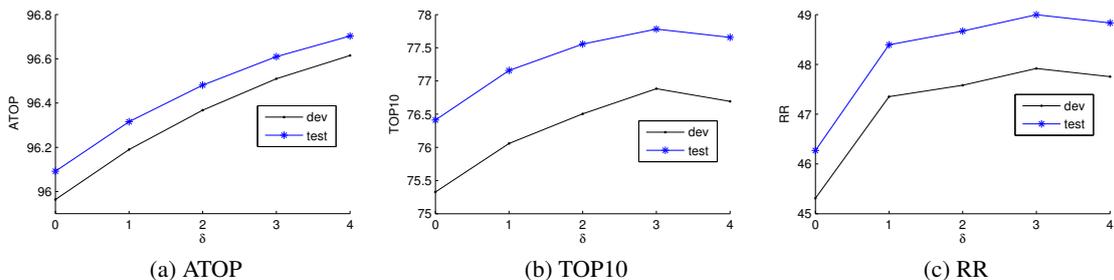


Figure 3: Impact of  $\delta$  ( $D = 100, k = 4$ )

score between the url referred news article and the tweet is compared against the scores of these 1,000 news articles to calculate the metric scores. 1/10 of the tweet/news pairs are used as development set, based on which all the parameters are tuned. The metrics ATOP, TOP10 and RR are used to evaluate the performance of systems.

## 6.2 Results

Table 1 summarizes the performance of the baselines and WTMF-G at latent dimension  $D = 100$ . All the parameters are chosen based on the development set. For WTMF-G, we try different values of  $k$  (the number of neighbors linked to a tweet/news for a hashtag/NE/time constraint) and  $\delta$  (the weight of link information). We choose to model the links in four subgraphs: (a) hashtags in tweet; (b) named entities in tweet; (c) time in tweet; (d) time in news article. For LDA we tune the hyperparameter  $\alpha$  (Dirichlet prior for topic distribution of a document) and  $\beta$  (Dirichlet prior for word distribution given a topic). It is worth noting that ATOP measures the overall ranking in 1000 samples while TOP10/RR focus on whether the aligned news article is in the first few returned results.

Same as reported in (Guo and Diab, 2012b), LDA- $\theta$  has the worst results due to directly using prior to the tweet, however, that will give a bias to some tweets, since the latter tweets have a larger candidate set than the earlier ones

the inferred topic distribution of a text  $\theta$ . The inferred topic vector has only a few non-zero values, hence a lot of information is missing. LDA-*wvec* preserves more information by creating a dense latent vector from the topic distribution of a word  $P(z|w)$ , and thus does much better in ATOP.

It is interesting to see that IR model has a very low ATOP (90.795%) and an acceptable RR (46.281%) score, in contrast to LDA-*wvec* with a high ATOP (94.148%) and a low RR(27.904%) score. This is caused by the nature of the two models. LDA-*wvec* is able to identify global coarse grained topic information (such as *politics* vs. *economics*), hence receiving a high ATOP by excluding the most irrelevant news articles, however it does not distinguish fine grained difference such as *Hillary* vs. *Obama*. IR model exerts the opposite influence via word matching. It ranks a correct news article very high if overlapping words exist (leading to a high RR), or the news article is ranked very low if no overlapping words (hence a low ATOP).

We can conclude WTMF is a very strong baseline given that it achieves high scores with three metrics. As a latent variable model, it is able to capture global topics (+1.89% ATOP over LDA-*wvec*); moreover, by explicitly modeling missing words, the existence of a word is also encoded in the latent vector (+2.31% TOP10 and -0.011% RR over IR model).

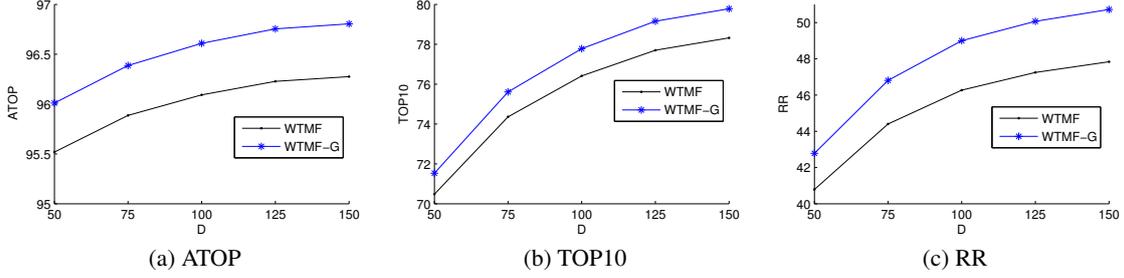


Figure 4: Impact of latent dimension  $D$  ( $k = 4$ )

Conditions	Links	ATOP		TOP10		RR	
		dev	test	dev	test	dev	test
hashtags_tweets	375,371	+0.397%	+0.379%	+1.015%	+1.021%	+0.504%	+0.641%
NE_tweets	164,412	+0.141%	+0.130%	+0.598%	+0.479%	+0.278%	+0.294%
time_tweet	139,488	+0.126%	+0.136%	+0.512%	+0.503%	+0.241%	+0.327%
time_news	50,008	+0.036%	+0.026%	+0.156%	+0.256%	+1.890%	+1.924%
full model (all 4 subgraphs)	573,999	<b>+0.546%</b>	<b>+0.518%</b>	<b>+1.556%</b>	<b>+1.371%</b>	<b>+2.607%</b>	<b>+2.727%</b>
full model <b>minus</b> hashtags_tweets	336,963	+0.288%	+0.276%	+1.129%	+1.037%	+2.488%	+2.541%
full model <b>minus</b> NE_tweets	536,333	+0.528%	+0.503%	+1.518%	<b>+1.393%</b>	+2.580%	+2.680%
full model <b>minus</b> time_tweet	466,207	+0.457%	+0.426%	+1.281%	+1.145%	+2.449%	+2.554%
full model <b>minus</b> time_news	523,991	+0.508%	+0.490%	+1.300%	+1.190%	+0.632%	+0.785%
author_tweet	21,318	+0.043%	+0.042%	+0.028%	+0.057%	-0.003%	-0.017%
full model <b>plus</b> author_tweet	593,483	+0.575%	+0.545%	+1.465%	+1.336%	+2.415%	+2.547%

Table 2: Contribution of subgraphs when  $D = 100$ ,  $k = 4$ ,  $\delta = 3$  (gain over baseline WTMF)

With WTMF being a very challenging baseline, WTMF-G can still significantly improve all 3 metrics. In the case  $k = 4$ ,  $\delta = 3$  compared to WTMF, WTMF-G receives +1.371% TOP10, +2.727% RR, and +0.518% ATOP value (this is a significant improvement of ATOP value considering that it is averaged on 30,000 data points, at an already high level of 96% reducing error rate by 13%). All the improvement of WTMF-G over WTMF is statistically significant at the 99% confidence level with a two-tailed paired t-test.

We also present results using different number of links  $k$  in WTMF-G in table 1. We experiment with  $k = \{3, 4, 5\}$ .  $k = 4$  is found to be the optimal value (although  $k = 5$  has a better ATOP). Figure 3 demonstrates the impact of  $\delta = \{0, 1, 2, 3, 4\}$  on each metric when  $k = 4$ . Note when  $\delta = 0$  no link is used, which is the baseline WTMF. We can see using links is always helpful. When  $\delta = 4$ , we receive a higher ATOP value but lower TOP10 and RR.

Figure 4 illustrates the impact of dimension  $D = \{50, 75, 100, 125, 150\}$  on WTMF and WTMF-G ( $k = 4$ ) on the test set. The trends hold in different  $D$  values with a consistent improvement. Generally a larger  $D$  leads to a better performance. In all conditions WTMF-G outperforms WTMF.

### 6.3 Contribution of Subgraphs

We are interested in the contribution of each feature subgraph. Therefore we list the impact of individual components in table 2. The impact of each subgraph is evaluated in two conditions: (a) the *subgraph-only*; (b) the *full-model-minus* the subgraph. The *full model* is the combination of the 4 subgraphs (which is also the best model  $k = 4$  in table 1). In the last two rows of table 2 we also present the results of using authorship only and the full model plus authorship. The 2nd column lists the number of links in the subgraph. To highlight the difference, we report the gain of each model over the baseline model WTMF.

We have several interesting observations from table 2. It is clear that the hashtag subgraph on tweets is the most useful subgraph: with hashtag\_tweet it has the best ATOP and TOP10 values among *subgraph-only* condition (ATOP: +0.379% vs. 2nd best +0.136%, TOP10: +1.021% vs. 2nd best +0.503%), while in the full-model-minus condition, minus hashtag has the lowest ATOP and TOP10. Observing that it also contains the most links, we believe the coverage is another important reason for the great performance.

It seems the named entity subgraph helps the least. Looking into the extracted named entities and hashtags, we find many popular named enti-

ties are covered by hashtags. That said, adding named entity subgraph into final model has a positive contribution.

It is worth noting that the `time_news` subgraph has the most positive influence on RR. This is because temporal information is very salient in news domain: usually there are several reports to describe an event within a short period, therefore the news latent vector is strengthened by receiving semantics from its neighbors.

At last, we analyze the influence of authorship of tweets. Adding authorship into the full model greatly hurts the scores of TOP10 and RR, whereas it is helpful to ATOP. This is understandable since by introducing author links between tweets, to some degree we are averaging the latent vectors of tweets written by the same person. Therefore, for a tweet whose topic is vague and hard to detect, it will get some prior knowledge of topics through the author links (hence increase ATOP), whereas this prior knowledge becomes noise for the tweets that are already handled very well by the model (hence decrease TOP10 and RR).

#### 6.4 Error Analysis

We look closely into ATOP results to obtain an intuitive feel for what is captured and what is not. For example, the ATOP score of WTMF for the tweet-news pair below is 89.9%:

*Tweet: ...stoked growing speculation that Pakistan's powerful military was quietly supporting moves... @declanwalsh*

*News: Pakistan Supreme Court Orders Arrest of Prime Minister*

By identifying “Pakistan” and “Supreme Court” as hashtags/named entity, WTMF-G is able to propagate the semantics from the following two informative tweets to the original tweet, hence achieving a higher ATOP score of 91.9%.

*#Pakistan Supreme Court orders the arrest of the PM on corruption charges.*

*A discouraging sign from a tumultuous political system: Pakistan's Supreme Court ordered the arrest of PM Ashraf today.*

Below is an example that shows the deficiency of both WTMF and WTMF-G:

*Tweet: Another reason to contemplate moving: an early death*

*News: America flunks its health exam*

In this case WTMF and WTMF-G achieve a

low ATOP of 69.8% and 75.1%, respectively. The only evidence the latent variable models rely on is lexical items (WTMF-G extract additional text-to-text correlation by word matching). To pinpoint the url referred news articles, other advanced NLP features should be exploited. In this case, we believe sentiment information could be helpful – both tweet and the news article contain a negative polarity.

## 7 Related Work

**Short Text Semantics:** The field of short text semantics has progressed immensely in recent years. Early work focus on word pair similarity in the high dimensional space. The word pair similarity is either knowledge based (Mihalcea et al., 2006; Tsatsaronis et al., 2010) or corpus based (Li et al., 2006; Islam and Inkpen, 2008), where co-occurrence information cannot be efficiently exploited. Guo and Diab (2012b; 2012a; 2013) show the superiority of the latent space approach in the WTMF model achieving state-of-the-art performance on two datasets. However, all of them only reply on text-to-word information. In this paper, we focus on modeling inter-text relations induced by Twitter/news features. We extend the WTMF model and adapt it into tweets modeling, achieving significantly better results.

**Modeling Tweets in a Latent Space:** Ramage et al. (2010) also use hashtags to improve the latent representation of tweets in a LDA framework, Labeled-LDA (Ramage et al., 2009), treating each hashtag as a label. Similar to the experiments presented in this paper, the result of using Labeled-LDA alone is worse than the IR model, due to the sparseness in the induced LDA latent vector. Jin et al. (2011) apply an LDA based model on clustering by incorporating url referred documents. The semantics of long documents are transferred to the topic distribution of tweets.

**News recommendation:** A news recommendation system aims to recommend news articles to a user based on the features (e.g., key words, tags, category) in the documents that the user likes (hence these documents form a training set) (Claypool et al., 1999; Corso et al., 2005; Lee and Park, 2007). Our paper resembles it in searching for a related news article. However, we target on recommending news article only based on a tweet, which is a much smaller context than the set of favorite documents chosen by a user .



- Weiwei Guo and Mona Diab. 2013. Improving lexical semantics for sentential semantics: Modeling selectional preference and similar words in a latent variable model. In *The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Hongzhao Huang, Arkaitz Zubiaga, Heng Ji, Hongbo Deng, Dong Wang, Hieu Le, Tarek Abdelzather, Jiwei Han, Alice Leung, John Hancock, and Clare Voss. 2012. Tweet ranking based on heterogeneous networks. In *Proceedings of the 24th International Conference on Computational Linguistics*.
- Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data*, 2.
- Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*.
- Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of Association for Computational Linguistics*.
- Ou Jin, Nathan N. Liu, Kai Zhao, Yong Yu, and Qiang Yang. 2011. Transferring topical knowledge from auxiliary long texts for short text clustering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*.
- Thomas K Landauer, Peter W. Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25.
- H. J. Lee and Sung Joo Park. 2007. Moners: A news recommender for the mobile web. *Expert Syst. Appl.*, 32(1):143–150.
- Yuhua Li, David McLean, Zuhair A. Bandar, James D. O’Shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transaction on Knowledge and Data Engineering*, 18.
- Hao Li, Yu Chen, Heng Ji, Smaranda Muresan, and Dequan Zheng. 2012. Combining social cognitive theories with linguistic features for multi-genre sentiment analysis. In *Proceedings of the 26th Pacific Asia Conference on Language, Information and Computation*.
- Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *The Semantic Web: Research and Applications*.
- Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence*.
- Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*.
- Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*.
- Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.
- Daniel Ramage, Susan Dumais, and Dan Liebling. 2010. Characterizing microblogs with topic models. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*.
- Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. 2009. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*.
- Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of Advances in Neural Information Processing Systems*.
- Michael Speriosui, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. 2011. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning*.
- Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL*.

George Tsatsaronis, Iraklis Varlamis, and Michalis Vazirgiannis. 2010. Text relatedness based on a word thesaurus. *Journal of Artificial Intelligence Research*, 37.

Andranik Tumasjan, Timm Oliver Sprenger, Philipp G. Sandner, and Isabell M. Welp. 2010. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *ICWSM*.

Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-markov continuous-time model of topical trends. In *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Rui Yan, Mirella Lapata, and Xiaoming Li. 2012. Tweet recommendation with graph co-ranking. In *Proceedings of the 24th International Conference on Computational Linguistics*.

Lei Yang, Tao Sun, Ming Zhang, and Qiaozhu Mei. 2012. We know what @you #tag: does the dual role affect hashtag adoption? In *Proceedings of the 21st international conference on World Wide Web*.

# A computational approach to politeness with application to social factors

Cristian Danescu-Niculescu-Mizil<sup>\*‡</sup>, Moritz Sudhof<sup>†</sup>, Dan Jurafsky<sup>†</sup>,  
Jure Leskovec<sup>\*</sup>, and Christopher Potts<sup>†</sup>

<sup>\*</sup>Computer Science Department, <sup>†</sup>Linguistics Department

<sup>\*†</sup>Stanford University, <sup>‡</sup>Max Planck Institute SWS

cristiand|jure@cs.stanford.edu, sudhof|jurafsky|cgpotts@stanford.edu

## Abstract

We propose a computational framework for identifying linguistic aspects of politeness. Our starting point is a new corpus of requests annotated for politeness, which we use to evaluate aspects of politeness theory and to uncover new interactions between politeness markers and context. These findings guide our construction of a classifier with domain-independent lexical and syntactic features operationalizing key components of politeness theory, such as *indirection*, *deference*, *impersonalization* and *modality*. Our classifier achieves close to human performance and is effective across domains. We use our framework to study the relationship between politeness and social power, showing that polite Wikipedia editors are more likely to achieve high status through elections, but, once elevated, they become less polite. We see a similar negative correlation between politeness and power on Stack Exchange, where users at the top of the reputation scale are less polite than those at the bottom. Finally, we apply our classifier to a preliminary analysis of politeness variation by gender and community.

## 1 Introduction

Politeness is a central force in communication, arguably as basic as the pressure to be truthful, informative, relevant, and clear (Grice, 1975; Leech, 1983; Brown and Levinson, 1978). Natural languages provide numerous and diverse means for encoding politeness and, in conversation, we constantly make choices about where and how to use these devices. Kaplan (1999) observes that “people desire to be *paid* respect” and identifies honorifics and other politeness markers, like *please*,

as “the coin of that payment”. In turn, politeness markers are intimately related to the power dynamics of social interactions and are often a decisive factor in whether those interactions go well or poorly (Gyasi Obeng, 1997; Chilton, 1990; Andersson and Pearson, 1999; Rogers and Lee-Wong, 2003; Holmes and Stubbe, 2005).

The present paper develops a computational framework for identifying and characterizing politeness marking in requests. We focus on requests because they involve the speaker imposing on the addressee, making them ideal for exploring the social value of politeness strategies (Clark and Schunk, 1980; Francik and Clark, 1985). Requests also stimulate extensive use of what Brown and Levinson (1987) call *negative politeness*: speaker strategies for minimizing (or appearing to minimize) the imposition on the addressee, for example, by being indirect (*Would you mind*) or apologizing for the imposition (*I’m terribly sorry, but*) (Lakoff, 1973; Lakoff, 1977; Brown and Levinson, 1978).

Our investigation is guided by a new corpus of requests annotated for politeness. The data come from two large online communities in which members frequently make requests of other members: Wikipedia, where the requests involve editing and other administrative functions, and Stack Exchange, where the requests center around a diverse range of topics (e.g., programming, gardening, cycling). The corpus confirms the broad outlines of linguistic theories of politeness pioneered by Brown and Levinson (1987), but it also reveals new interactions between politeness markings and the morphosyntactic context. For example, the politeness of *please* depends on its syntactic position and the politeness markers it co-occurs with.

Using this corpus, we construct a politeness classifier with a wide range of domain-independent lexical, sentiment, and dependency features operationalizing key components of po-

liteness theory, including not only the negative politeness markers mentioned above but also elements of *positive politeness* (gratitude, positive and optimistic sentiment, solidarity, and inclusiveness). The classifier achieves near human-level accuracy across domains, which highlights the consistent nature of politeness strategies and paves the way to using the classifier to study new data.

Politeness theory predicts a negative correlation between politeness and the power of the requester, where power is broadly construed to include social status, authority, and autonomy (Brown and Levinson, 1987). The greater the speaker’s power relative to her addressee, the less polite her requests are expected to be: there is no need for her to incur the expense of paying respect, and failing to make such payments can invoke, and hence reinforce, her power. We support this prediction by applying our politeness framework to Wikipedia and Stack Exchange, both of which provide independent measures of social status. We show that polite Wikipedia editors are more likely to achieve high status through elections; however, once elected, they become less polite. Similarly, on Stack Exchange, we find that users at the top of the reputation scale are less polite than those at the bottom.

Finally, we briefly address the question of how politeness norms vary across communities and social groups. Our findings confirm established results about the relationship between politeness and gender, and they identify substantial variation in politeness across different programming language subcommunities on Stack Exchange.

## 2 Politeness data

Requests involve an imposition on the addressee, making them a natural domain for studying the inter-connections between linguistic aspects of politeness and social variables.

**Requests in online communities** We base our analysis on two online communities where requests have an important role: the Wikipedia community of editors and the Stack Exchange question-answer community.<sup>1</sup> On Wikipedia, to coordinate on the creation and maintenance of the collaborative encyclopedia, editors can interact with each other on user talk-pages;<sup>2</sup> re-

<sup>1</sup><http://stackexchange.com/about>

<sup>2</sup>[http://en.wikipedia.org/wiki/Wikipedia:User\\_pages](http://en.wikipedia.org/wiki/Wikipedia:User_pages)

quests posted on a user talk-page, although public, are generally directed to the owner of the talk-page. On Stack Exchange, users often comment on existing posts requesting further information or proposing edits; these requests are generally directed to the authors of the original posts.

Both communities are not only rich in user-to-user requests, but these requests are also part of consequential conversations, not empty social banter; they solicit specific information or concrete actions, and they expect a response.

**Politeness annotation** Computational studies of politeness, or indeed any aspect of linguistic pragmatics, demand richly labeled data. We therefore label a large portion of our request data (over 10,000 utterances) using Amazon Mechanical Turk (AMT), creating the largest corpus with politeness annotations (see Table 1 for details).<sup>3</sup>

We choose to annotate requests containing exactly two sentences, where the second sentence is the actual request (and ends with a question mark). This provides enough context to the annotators while also controlling for length effects. Each annotator was instructed to read a batch of 13 requests and consider them as originating from a co-worker by email. For each request, the annotator had to indicate how polite she perceived the request to be by using a slider with values ranging from “very impolite” to “very polite”.<sup>4</sup> Each request was labeled by five different annotators.

We vetted annotators by restricting their residence to be in the U.S. and by conducting a linguistic background questionnaire. We also gave them a paraphrasing task shown to be effective for verifying and eliciting linguistic attentiveness (Munro et al., 2010), and we monitored the annotation job and manually filtered out annotators who submitted uniform or seemingly random annotations.

Because politeness is highly subjective and annotators may have inconsistent scales, we applied the standard z-score normalization to each worker’s scores. Finally, we define the politeness score (henceforth *politeness*) of a request as the average of the five scores assigned by the annotators. The distribution of resulting request scores (shown in Figure 1) has an average of 0 and stan-

<sup>3</sup>Publicly available at <http://www.mpi-sws.org/~cristian/Politeness.html>

<sup>4</sup>We used non-categorical ratings for finer granularity and to help account for annotators’ different perception scales.

domain	#requests	#annotated	#annotators
Wiki	35,661	4,353	219
SE	373,519	6,604	212

Table 1: Summary of the request data and its politeness annotations.

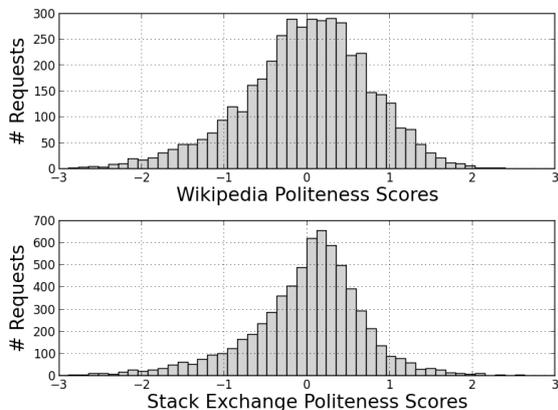


Figure 1: Distribution of politeness scores. Positive scores indicate requests perceived as polite.

standard deviation of 0.7 for both domains; positive values correspond to polite requests (i.e., requests with normalized annotations towards the “very polite” extreme) and negative values to impolite requests. A summary of all our request data is shown in Table 1.

**Inter-annotator agreement** To evaluate the reliability of the annotations we measure the inter-annotator agreement by computing, for each batch of 13 documents that were annotated by the same set of 5 users, the mean pairwise correlation of the respective scores. For reference, we compute the same quantities after randomizing the scores by sampling from the observed distribution of politeness scores. As shown in Figure 2, the labels are coherent and significantly different from the randomized procedure ( $p < 0.0001$  according to a Wilcoxon signed rank test).<sup>5</sup>

**Binary perception** Although we did not impose a discrete categorization of politeness, we acknowledge an implicit binary perception of the phenomenon: whenever an annotator moved a slider in one direction or the other, she made a binary politeness judgment. However, the bound-

<sup>5</sup>The commonly used Cohen/Fleiss Kappa agreement measures are not suitable for this type of annotation, in which labels are continuous rather than categorical.

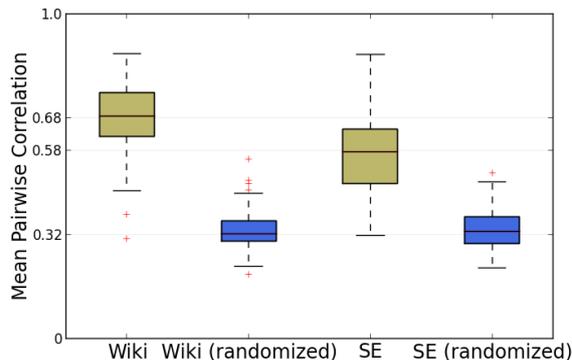


Figure 2: Inter-annotator pairwise correlation, compared to the same measure after randomizing the scores.

Quartile:	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	4 <sup>th</sup>
Wiki	62%	8%	3%	51%
SE	37%	4%	6%	46%

Table 2: The percentage of requests for which all five annotators agree on binary politeness. The 4<sup>th</sup> quartile contains the requests with the top 25% politeness scores in the data. (For reference, randomized scoring yields agreement percentages of <20% for all quartiles.)

ary between somewhat polite and somewhat impolite requests can be blurry. To test this intuition, we break the set of annotated requests into four groups, each corresponding to a politeness score quartile. For each quartile, we compute the percentage of requests for which all five annotators made the same binary politeness judgment. As shown in Table 2, full agreement is much more common in the 1<sup>st</sup> (bottom) and 4<sup>th</sup> (top) quartiles than in the middle quartiles. This suggests that the politeness scores assigned to requests that are only somewhat polite or somewhat impolite are less reliable and less tied to an intuitive notion of binary politeness. This discrepancy motivates our choice of classes in the prediction experiments (Section 4) and our use of the top politeness quartile (the 25% most polite requests) as a reference in our subsequent discussion.

### 3 Politeness strategies

As we mentioned earlier, requests impose on the addressee, potentially placing her in social peril if she is unwilling or unable to comply. Requests therefore naturally give rise to the negative po-

liteness strategies of Brown and Levinson (1987), which are attempts to mitigate these social threats. These strategies are prominent in Table 3, which describes the core politeness markers we analyzed in our corpus of Wikipedia requests. We do not include the Stack Exchange data in this analysis, reserving it as a “test community” for our prediction task (Section 4).

Requests exhibiting politeness markers are automatically extracted using regular expression matching on the dependency parse obtained by the Stanford Dependency Parser (de Marneffe et al., 2006), together with specialized lexicons. For example, for the hedges marker (Table 3, line 19), we match all requests containing a nominal subject dependency edge pointing out from a hedge verb from the hedge list created by Hyland (2005). For each politeness strategy, Table 3 shows the average **politeness** score of the respective requests (as described in Section 2; positive numbers indicate polite requests), and their **top politeness quartile** membership (i.e., what percentage fall within the top quartile of politeness scores). As discussed at the end of Section 2, the top politeness quartile gives a more robust and more intuitive measure of politeness. For reference, a random sample of requests will have a 0 politeness score and a 25% top quartile membership; in both cases, larger numbers indicate higher politeness.

Gratitude and deference (lines 1–2) are ways for the speaker to incur a social cost, helping to balance out the burden the request places on the addressee. Adopting Kaplan (1999)’s metaphor, these are the coin of the realm when it comes to paying the addressee respect. Thus, they are indicators of positive politeness.

Terms from the sentiment lexicon (Liu et al., 2005) are also tools for positive politeness, either by emphasizing a positive relationship with the addressee (line 4), or being impolite by using negative sentiment that damages this positive relationship (line 5). Greetings (line 3) are another way to build a positive relationship with the addressee.

The remainder of the cues in Table 3 are negative politeness strategies, serving the purpose of minimizing, at least in appearance, the imposition on the addressee. Apologizing (line 6) deflects the social threat of the request by attuning to the imposition itself. Being indirect (line 9) is another way to minimize social threat. This strategy allows the speaker to avoid words and phrases convention-

ally associated with requests. First-person plural forms like *we* and *our* (line 15) are also ways of being indirect, as they create the sense that the burden of the request is shared between speaker and addressee (*We really should . . .*). Though indirectness is not invariably interpreted as politeness marking (Blum-Kulka, 2003), it is nonetheless a reliable marker of it, as our scores indicate. What’s more, direct variants (imperatives, statements about the addressee’s obligations) are less polite (lines 10–11).

Indirect strategies also combine with hedges (line 19) conveying that the addressee is unlikely to accept the burden (*Would you by any chance . . . ?*, *Would it be at all possible . . . ?*). These too serve to provide the addressee with a face-saving way to deny the request. We even see subtle effects of modality at work here: the irrealis, counterfactual forms *would* and *could* are more polite than their ability (dispositional) or future-oriented variants *can* and *will*; compare lines 12 and 13. This parallels the contrast between factuality markers (impolite; line 20) and hedging (polite; line 19).

Many of these features are correlated with each other, in keeping with the insight of Brown and Levinson (1987) that politeness markers are often combined to create a cumulative effect of increased politeness. Our corpora also highlight interactions that are unexpected (or at least unaccounted for) on existing theories of politeness. For example, sentence-medial *please* is polite (line 7), presumably because of its freedom to combine with other negative politeness strategies (*Could you please . . .*). In contrast, sentence-initial *please* is impolite (line 8), because it typically signals a more direct strategy (*Please do this*), which can make the politeness marker itself seem insincere. We see similar interactions between pronominal forms and syntactic structure: sentence-initial *you* is impolite (*You need to . . .*), whereas sentence-medial *you* is often part of the indirect strategies we discussed above (*Would/Could you . . .*).

#### 4 Predicting politeness

We now show how our linguistic analysis can be used in a machine learning model for automatically classifying requests according to politeness. A classifier can help verify the predictive power, robustness, and domain-independent generality of the linguistic strategies of Section 3. Also, by providing automatic politeness judgments for large

Strategy	Politeness	In top quartile	Example
1. Gratitude	0.87 <sup>***</sup>	78% <sup>***</sup>	<b>I</b> really <b>appreciate</b> that you’ve done them.
2. Deference	0.78 <sup>***</sup>	70% <sup>***</sup>	<b>Nice work</b> so far on your rewrite.
3. Greeting	0.43 <sup>***</sup>	45% <sup>***</sup>	<b>Hey</b> , I just tried to ...
4. Positive lexicon	0.12 <sup>***</sup>	32% <sup>***</sup>	<b>Wow!</b> / This is a <b>great</b> way to deal. ...
5. Negative lexicon	-0.13 <sup>***</sup>	22% <sup>**</sup>	If you’re going to <b>accuse</b> me ...
6. Apologizing	0.36 <sup>***</sup>	53% <sup>***</sup>	<b>Sorry</b> to bother you ...
7. Please	0.49 <sup>***</sup>	57% <sup>***</sup>	Could you <b>please</b> say more. ...
8. Please start	-0.30 <sup>*</sup>	22%	<b>Please</b> do not remove warnings ...
9. Indirect (btw)	0.63 <sup>***</sup>	58% <sup>**</sup>	<b>By the way</b> , where did you find ...
10. Direct question	-0.27 <sup>***</sup>	15% <sup>***</sup>	<b>What</b> is your native language?
11. Direct start	-0.43 <sup>***</sup>	9% <sup>***</sup>	<b>So</b> can you retrieve it or not?
12. Counterfactual modal	0.47 <sup>***</sup>	52% <sup>***</sup>	<b>Could/Would</b> you ...
13. Indicative modal	0.09	27%	<b>Can/Will</b> you ...
14. 1st person start	0.12 <sup>***</sup>	29% <sup>**</sup>	<b>I</b> have just put the article ...
15. 1st person pl.	0.08 <sup>*</sup>	27%	Could <b>we</b> find a less complex name ...
16. 1st person	0.08 <sup>***</sup>	28% <sup>***</sup>	It is <b>my</b> view that ...
17. 2nd person	0.05 <sup>***</sup>	30% <sup>***</sup>	But what’s the good source <b>you</b> have in mind?
18. 2nd person start	-0.30 <sup>***</sup>	17% <sup>**</sup>	<b>You</b> ’ve reverted yourself ...
19. Hedges	0.14 <sup>***</sup>	28%	<b>I suggest</b> we start with ...
20. Factuality	-0.38 <sup>***</sup>	13% <sup>***</sup>	<b>In fact</b> you did link, ...

Table 3: Positive (1-5) and negative (6–20) politeness strategies and their relation to human perception of politeness. For each strategy we show the average (human annotated) **politeness** scores for the requests exhibiting that strategy (compare with 0 for a random sample of requests; a positive number indicates the strategy is perceived as being polite), as well as the percentage of requests exhibiting the respective strategy that fall in the **top quartile** of politeness scores (compare with 25% for a random sample of requests). Throughout the paper: for politeness scores, statistical significance is calculated by comparing the set of requests exhibiting the strategy with the rest using a Mann-Whitney-Wilcoxon U test; for top quartile membership a binomial test is used.

amounts of new data on a scale unfeasible for human annotation, it can also enable a detailed analysis of the relation between politeness and social factors (Section 5).

**Task setup** To evaluate the robustness and domain-independence of the analysis from Section 3, we run our prediction experiments on two very different domains. We treat Wikipedia as a “development domain” since we used it for developing and identifying features and for training our models. Stack Exchange is our “test domain” since it was not used for identifying features. We take the model (features and weights) trained on Wikipedia and use them to classify requests from Stack Exchange.

We consider two classes of requests: **polite** and **impolite**, defined as the top and, respectively, bottom quartile of requests when sorted by their politeness score (based on the binary notion of politeness discussed in Section 2). The classes are therefore balanced, with each class consisting of 1,089 requests for the Wikipedia domain and 1,651 requests for the Stack Exchange domain.

We compare two classifiers — a bag of words classifier (BOW) and a linguistically informed classifier (Ling.) — and use human labelers as a reference point. The BOW classifier is an SVM using a unigram feature representation.<sup>6</sup> We consider this to be a strong baseline for this new

<sup>6</sup>Unigrams appearing less than 10 times are excluded.

classification task, especially considering the large amount of training data available. The linguistically informed classifier (Ling.) is an SVM using the linguistic features listed in Table 3 in addition to the unigram features. Finally, to obtain a reference point for the prediction task we also collect three new politeness annotations for each of the requests in our dataset using the same methodology described in Section 2. We then calculate human performance on the task (Human) as the percentage of requests for which the average score from the additional annotations matches the binary politeness class of the original annotations (e.g., a positive score corresponds to the polite class).

**Classification results** We evaluate the classifiers both in an in-domain setting, with a standard leave-one-out cross validation procedure, and in a cross-domain setting, where we train on one domain and test on the other (Table 4). For both our development and our test domains, and in both the in-domain and cross-domain settings, the linguistically informed features give 3-4% absolute improvement over the bag of words model. While the in-domain results are within 3% of human performance, the greater room for improvement in the cross-domain setting motivates further research on linguistic cues of politeness.

The experiments in this section confirm that our theory-inspired features are indeed effective in practice, and generalize well to new domains. In the next section we exploit this insight to automatically annotate a much larger set of requests (about 400,000) with politeness labels, enabling us to relate politeness to several social variables and outcomes. For new requests, we use class probability estimates obtained by fitting a logistic regression model to the output of the SVM (Witten and Frank, 2005) as *predicted politeness scores* (with values between 0 and 1; henceforth *politeness*, by abuse of language).

## 5 Relation to social factors

We now apply our framework to studying the relationship between politeness and social variables, focussing on social power dynamics. Encouraged by the close-to-human performance of our in-domain classifiers, we use them to assign politeness labels to our full dataset and then compare these labels to independent measures of power and status in our data. The results closely match those obtained with human-labeled data alone, thereby

Train Test	In-domain		Cross-domain	
	Wiki	SE	Wiki	SE
BOW	79.84%	74.47%	64.23%	72.17%
Ling.	83.79%	78.19%	67.53%	75.43%
Human	86.72%	80.89%	80.89%	86.72%

Table 4: Accuracies of our two classifiers for Wikipedia (Wiki) and Stack Exchange (SE), for in-domain and cross-domain settings. Human performance is included as a reference point. The random baseline performance is 50%.

supporting the use of computational methods to pursue questions about social variables.

### 5.1 Relation to social outcome

Earlier, we characterized politeness markings as currency used to pay respect. Such language is therefore costly in a social sense, and, relatedly, tends to incur costs in terms of communicative efficiency (Van Rooy, 2003). Are these costs worth paying? We now address this question by studying politeness in the context of the electoral system of the Wikipedia community of editors.

Among Wikipedia editors, status is a salient social variable (Anderson et al., 2012). Administrators (*admins*) are editors who have been granted certain rights, including the ability to block other editors and to protect or delete articles.<sup>7</sup> Admins have a higher status than common editors (*non-admins*), and this distinction seems to be widely acknowledged by the community (Burke and Kraut, 2008b; Leskovec et al., 2010; Danescu-Niculescu-Mizil et al., 2012). Aspiring editors become admins through public elections,<sup>8</sup> so we know when the status change from non-admin to admins occurred and can study users' language use in relation to that time.

To see whether politeness correlates with eventual high status, we compare, in Table 5, the politeness levels of requests made by users who will eventually succeed in becoming administrators (Eventual status: Admins) with requests made by users who are not admins (Non-admins).<sup>9</sup> We observe that admins-to-be are significantly more po-

<sup>7</sup><http://en.wikipedia.org/wiki/Wikipedia:Administrators>

<sup>8</sup>[http://en.wikipedia.org/wiki/Wikipedia:Requests\\_for\\_adminship](http://en.wikipedia.org/wiki/Wikipedia:Requests_for_adminship)

<sup>9</sup>We consider only requests made up to one month before the election, to avoid confusion with pre-election behavior.

Eventual status	Politeness	Top quart.
Admins	0.46**	30%***
Non-admins	0.39***	25%
Failed	0.37**	22%

Table 5: Politeness and status. Editors who will eventually become admins are more polite than non-admins ( $p < 0.001$  according to a Mann-Whitney-Wilcoxon U test) and than editors who will eventually fail to become admins ( $p < 0.001$ ). Out of their requests, 30% are rated in the top politeness quartile (significantly more than the 25% of a random sample;  $p < 0.001$  according to a binomial test). This analysis was conducted on 31k requests (1.4k for Admins, 28.9k for Non-admins, 652 for Failed).

lite than non-admins. One might wonder whether this merely reflects the fact that not all users aspire to become admins, and those that do are more polite. To address this, we also consider users who ran for adminship but did not earn community approval (Eventual status: Failed). These users are also significantly less polite than their successful counterparts, indicating that politeness indeed correlates with a positive social outcome here.

## 5.2 Politeness and power

We expect a rise in status to correlate with a decline in politeness (as predicted by politeness theory, and discussed in Section 1). The previous section does not test this hypothesis, since all editors compared in Table 5 had the same (non-admin) status when writing the requests. However, our data does provide three ways of testing this hypothesis.

First, after the adminship elections, successful editors get a boost in power by receiving admin privileges. Figure 3 shows that this boost is mirrored by a significant decrease in politeness (blue, diamond markers). Losing an election has the opposite effect on politeness (red, circle markers), perhaps as a consequence of reinforced low status.

Second, Stack Exchange allows us to test more situational power effects.<sup>10</sup> On the site, users request, from the community, information they are lacking. This informational asymmetry between the question-asker and his audience puts him at

<sup>10</sup>We restrict all experiments in this section to the largest subcommunity of Stack Exchange, namely Stack Overflow.

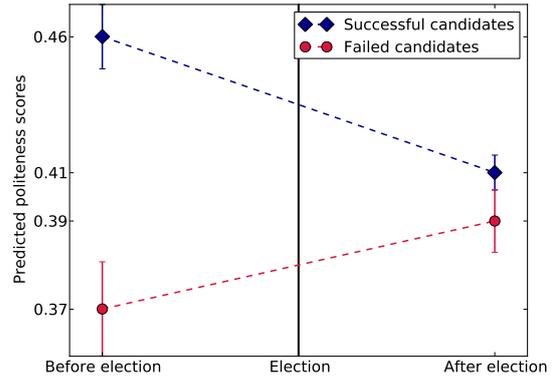


Figure 3: Successful and failed candidates before and after elections. Editors that will eventually succeed (diamond marker) are significantly more polite than those that will fail (circle markers). Following the elections, successful editors become less polite while unsuccessful editors become more polite.

a social disadvantage. We therefore expect the question-asker to be more polite than the people who respond. Table 6 shows that this expectation is born out: comments posted to a thread by the original question-asker are more polite than those posted by other users.

Role	Politeness	Top quart.
Question-asker	0.65***	32%***
Answer-givers	0.52***	20%***

Table 6: Politeness and dependence. Requests made in comments posted by the question-asker are significantly more polite than the other requests. Analysis conducted on 181k requests (106k for question-askers, 75k for answer-givers).

Third, Stack Exchange allows us to examine power in the form of authority, through the community’s reputation system. Again, we see a negative correlation between politeness and power, even after controlling for the role of the user making the requests (i.e., Question-asker or Answer-giver). Table 7 summarizes the results.<sup>11</sup>

**Human validation** The above analyses are based on predicted politeness from our classifier. This allows us to use the entire request data cor-

<sup>11</sup>Since our data does not contain time stamps for reputation scores, we only consider requests that were issued in the six months prior to the available snapshot.

Reputation level	Politeness	Top quart.
Low reputation	0.68***	27%***
Middle reputation	0.66***	25%
High reputation	0.64***	23%***

Table 7: Politeness and Stack Exchange reputation (texts by question-askers only). High-reputation users are less polite. Analysis conducted on 25k requests (4.5k low, 12.5k middle, 8.4k high).

pus to test our hypotheses and to apply precise controls to our experiments (such as restricting our analysis to question-askers in the reputation experiment). In order to validate this methodology, we turned again to human annotation: we collected additional politeness annotation for the types of requests involved in the newly designed experiments. When we re-ran our experiments on human-labeled data alone we obtained the same qualitative results, with statistical significance always lower than 0.01.<sup>12</sup>

**Prediction-based interactions** The human validation of classifier-based results suggests that our prediction framework can be used to explore differences in politeness levels across factors of interest, such as communities, geographical regions and gender, even where gathering sufficient human-annotated data is infeasible. We mention just a few such preliminary results here: (i) Wikipedians from the U.S. Midwest are most polite (when compared to other census-defined regions), (ii) female Wikipedians are generally more polite (consistent with prior studies in which women are more polite in a variety of domains; (Herring, 1994)), and (iii) programming language communities on Stack Exchange vary significantly by politeness (Table 8; full disclosure: our analyses were conducted in Python).

## 6 Related work

Politeness has been a central concern of modern pragmatic theory since its inception (Grice, 1975; Lakoff, 1973; Lakoff, 1977; Leech, 1983; Brown and Levinson, 1978), because it is a source of pragmatic enrichment, social meaning, and cultural variation (Harada, 1976; Matsumoto, 1988;

<sup>12</sup>However, due to the limited size of the human-labeled data, we could not control for the role of the user in the Stack Exchange reputation experiment.

PL name	Politeness	Top quartile
Python	0.47***	23%
Perl	0.49	24%
PHP	0.51	24%
Javascript	0.53**	26%**
Ruby	0.59***	28%*

Table 8: Politeness of requests from different language communities on Stack Exchange.

Ide, 1989; Blum-Kulka and Kasper, 1990; Blum-Kulka, 2003; Watts, 2003; Byon, 2006). The starting point for most research is the theory of Brown and Levinson (1987). Aspects of this theory have been explored from game-theoretic perspectives (Van Rooy, 2003) and implemented in language generation systems for interactive narratives (Walker et al., 1997), cooking instructions, (Gupta et al., 2007), translation (Faruqui and Pado, 2012), spoken dialog (Wang et al., 2012), and subjectivity analysis (Abdul-Mageed and Diab, 2012), among others.

In recent years, politeness has been studied in online settings. Researchers have identified variation in politeness marking across different contexts and media types (Herring, 1994; Brennan and Ohaeri, 1999; Duthler, 2006) and between different social groups (Burke and Kraut, 2008a). The present paper pursues similar goals using orders of magnitude more data, which facilitates a fuller survey of different politeness strategies.

Politeness marking is one aspect of the broader issue of how language relates to power and status, which has been studied in the context of workplace discourse (Bramsen et al., ; Diehl et al., 2007; Peterson et al., 2011; Prabhakaran et al., 2012; Gilbert, 2012; McCallum et al., 2007) and social networking (Scholand et al., 2010). However, this research focusses on domain-specific textual cues, whereas the present work seeks to leverage domain-independent politeness cues, building on the literature on how politeness affects workplace social dynamics and power structures (Gyasi Obeng, 1997; Chilton, 1990; Andersson and Pearson, 1999; Rogers and Lee-Wong, 2003; Holmes and Stubbe, 2005). Burke and Kraut (2008b) study the question of how and why specific individuals rise to administrative positions on Wikipedia, and Danescu-Niculescu-Mizil et al. (2012) show that power differences on Wikipedia

are revealed through aspects of linguistic accommodation. The present paper complements this work by revealing the role of politeness in social outcomes and power relations.

## 7 Conclusion

We construct and release a large collection of politeness-annotated requests and use it to evaluate key aspects of politeness theory. We build a politeness classifier that achieves near-human performance and use it to explore the relation between politeness and social factors such as power, status, gender, and community membership. We hope the publicly available collection of annotated requests enables further study of politeness and its relation to social factors, as this paper has only begun to explore this area.

## Acknowledgments

We thank Jean Wu for running the AMT annotation task, and all the participating turkers. We thank Diana Minculescu and the anonymous reviewers for their helpful comments. This work was supported in part by NSF IIS-1016909, CNS-1010921, IIS-1149837, IIS-1159679, ARO MURI, DARPA SMISC, Okawa Foundation, Docomo, Boeing, Allyes, Volkswagen, Intel, Alfred P. Sloan Fellowship, the Microsoft Faculty Fellowship, the Gordon and Dailey Pattee Faculty Fellowship, and the Center for Advanced Study in the Behavioral Sciences at Stanford.

## References

- Muhammad Abdul-Mageed and Mona Diab. 2012. AWATIF: A multi-genre corpus for Modern Standard Arabic subjectivity and sentiment analysis. In *Proceedings of LREC*, pages 3907–3914.
- Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Effects of user similarity in social media. In *Proceedings of WSDM*, pages 703–712.
- Lynne M. Andersson and Christine M. Pearson. 1999. Tit for tat? the spiraling effect of incivility in the workplace. *The Academy of Management Review*, 24(3):452–471.
- Shoshana Blum-Kulka and Gabriele Kasper. 1990. Special issue on politeness. *Journal of Pragmatics*, 144(2).
- Shoshana Blum-Kulka. 2003. Indirectness and politeness in requests: Same or different? *Journal of Pragmatics*, 11(2):131–146.
- Philip Bramsen, Martha Escobar-Molana, Ami Patel, and Rafael Alonso. Extracting social power relationships from natural language. In *Proceedings of ACL*, pages 773–782.
- Susan E Brennan and Justina O Ohaeri. 1999. Why do electronic conversations seem less polite? the costs and benefits of hedging. *SIGSOFT Softw. Eng. Notes*, 24(2):227–235.
- Penelope Brown and Stephen C. Levinson. 1978. Universals in language use: Politeness phenomena. In Esther N. Goody, editor, *Questions and Politeness: Strategies in Social Interaction*, pages 56–311, Cambridge. Cambridge University Press.
- Penelope Brown and Stephen C Levinson. 1987. *Politeness: some universals in language usage*. Cambridge University Press.
- Moira Burke and Robert Kraut. 2008a. Mind your Ps and Qs: the impact of politeness and rudeness in online communities. In *Proceedings of CSCW*, pages 281–284.
- Moira Burke and Robert Kraut. 2008b. Taking up the mop: identifying future wikipedia administrators. In *CHI '08 extended abstracts on Human factors in computing systems*, pages 3441–3446.
- Andrew Sangpil Byon. 2006. The role of linguistic indirectness and honorifics in achieving linguistic politeness in Korean requests. *Journal of Politeness Research*, 2(2):247–276.
- Paul Chilton. 1990. Politeness, politics, and diplomacy. *Discourse and Society*, 1(2):201–224.
- Herbert H. Clark and Dale H. Schunk. 1980. Polite responses to polite requests. *Cognition*, 8(1):111–143.
- Cristian Danescu-Niculescu-Mizil, Lillian Lee, Bo Pang, and Jon Kleinberg. 2012. Echoes of power: Language effects and power differences in social interaction. In *Proceedings of WWW*, pages 699–708.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, pages 449–454.
- Christopher P. Diehl, Galileo Namata, and Lise Getoor. 2007. Relationship identification for social network discovery. In *Proceedings of the AAAI Workshop on Enhanced Messaging*, pages 546–552.
- Kirk W Duthler. 2006. The Politeness of Requests Made Via Email and Voicemail: Support for the Hyperpersonal Model. *Journal of Computer-Mediated Communication*, 11(2):500–521.
- Manaal Faruqui and Sebastian Pado. 2012. Towards a model of formal and informal address in english. In *Proceedings of EACL*, pages 623–633.

- Elen P. Francik and Herbert H. Clark. 1985. How to make requests that overcome obstacles to compliance. *Journal of Memory and Language*, 24:560–568.
- Eric Gilbert. 2012. Phrases that signal workplace hierarchy. In *Proceedings of CSCW*, pages 1037–1046.
- H. Paul Grice. 1975. Logic and conversation. In Peter Cole and Jerry Morgan, editors, *Syntax and Semantics*, volume 3: Speech Acts, pages 43–58. Academic Press, New York.
- S Gupta, M Walker, and D Romano. 2007. How rude are you?: Evaluating politeness and affect in interaction. *Affective Computing and Intelligent Interaction*, pages 203–217.
- Samuel Gyasi Obeng. 1997. Language and politics: Indirectness in political discourse. *Discourse and Society*, 8(1):49–83.
- S. I. Harada. 1976. Honorifics. In Masayoshi Shibatani, editor, *Syntax and Semantics*, volume 5: Japanese Generative Grammar, pages 499–561. Academic Press, New York.
- Susan Herring. 1994. Politeness in computer culture: Why women thank and men flame. In *Cultural performances: Proceedings of the third Berkeley women and language conference*, volume 278, page 94.
- Janet Holmes and Maria Stubbe. 2005. *Power and Politeness in the Workplace: A Sociolinguistic Analysis of Talk at Work*. Longman, London.
- Ken Hyland. 2005. *Metadiscourse: Exploring Interaction in Writing*. Continuum, London and New York.
- Sachiko Ide. 1989. Formal forms and discernment: Two neglected aspects of universals of linguistic politeness. *Multilingua*, 8(2–3):223–248.
- David Kaplan. 1999. What is meaning? Explorations in the theory of *Meaning as Use*. Brief version — draft 1. Ms., UCLA.
- Robin Lakoff. 1973. The logic of politeness; or, minding your P's and Q's. In *Proceedings of the 9th Meeting of the Chicago Linguistic Society*, pages 292–305.
- Robin Lakoff. 1977. What you can do with words: Politeness, pragmatics and performatives. In *Proceedings of the Texas Conference on Performatives, Presuppositions and Implicatures*, pages 79–106.
- Geoffrey N. Leech. 1983. *Principles of Pragmatics*. Longman, London and New York.
- Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Governance in Social Media: A case study of the Wikipedia promotion process. In *Proceedings of ICWSM*, pages 98–105.
- Bing Liu, Mingqing Hu, and Junsheng Cheng. 2005. Opinion Observer: analyzing and comparing opinions on the Web. In *Proceedings of WWW*, pages 342–351.
- Yoshiko Matsumoto. 1988. Reexamination of the universality of face: Politeness phenomena in Japanese. *Journal of Pragmatics*, 12(4):403–426.
- Andrew McCallum, Xuerui Wang, and Andr'es Corrada-Emmanuel. 2007. Topic and role discovery in social networks with experiments on Enron and academic email. *Journal of Artificial Intelligence Research*, 30(1):249–272.
- Robert Munro, Steven Bethard, Victor Kuperman, Vicky Tzuyin Lai, Robin Melnick, Christopher Potts, Tyler Schnoebelen, and Harry Tily. 2010. Crowdsourcing and language studies: the new generation of linguistic data. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 122–130.
- Kelly Peterson, Matt Hohensee, and Fei Xia. 2011. Email formality in the workplace: A case study on the enron corpus. In *Proceedings of the ACL Workshop on Language in Social Media*, pages 86–95.
- Vinodkumar Prabhakaran, Owen Rambow, and Mona Diab. 2012. Predicting Overt Display of Power in Written Dialogs. In *Proceedings of NAACL-HLT*, pages 518–522.
- Priscilla S. Rogers and Song Mei Lee-Wong. 2003. Reconceptualizing politeness to accommodate dynamic tensions in subordinate-to-superior reporting. *Journal of Business and Technical Communication*, 17(4):379–412.
- Andrew J. Scholand, Yla R. Tausczik, and James W. Pennebaker. 2010. Social language network analysis. In *Proceedings of CSCW*, pages 23–26.
- Robert Van Rooy. 2003. Being polite is a handicap: Towards a game theoretical analysis of polite linguistic behavior. In *Proceedings of TARK*, pages 45–58.
- Marilyn A Walker, Janet E Cahn, and Stephen J Whitaker. 1997. Improvising linguistic style: social and affective bases for agent personality. In *Proceedings of AGENTS*, pages 96–105.
- William Yang Wang, Samantha Finkelstein, Amy Ogan, Alan W. Black, and Justine Cassell. 2012. "love ya, jerkface": Using sparse log-linear models to build positive and impolite relationships with teens. In *Proceedings of SIGDIAL*, pages 20–29.
- Richard J. Watts. 2003. *Politeness*. Cambridge University Press, Cambridge.
- Ian H Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

# Modeling Thesis Clarity in Student Essays

Isaac Persing and Vincent Ng

Human Language Technology Research Institute

University of Texas at Dallas

Richardson, TX 75083-0688

{persingq, vince}@hlt.utdallas.edu

## Abstract

Recently, researchers have begun exploring methods of scoring student essays with respect to particular dimensions of quality such as coherence, technical errors, and relevance to prompt, but there is relatively little work on modeling thesis clarity. We present a new annotated corpus and propose a learning-based approach to scoring essays along the thesis clarity dimension. Additionally, in order to provide more valuable feedback on why an essay is scored as it is, we propose a second learning-based approach to identifying what kinds of errors an essay has that may lower its thesis clarity score.

## 1 Introduction

Automated essay scoring, the task of employing computer technology to evaluate and score written text, is one of the most important educational applications of natural language processing (NLP) (see Shermis and Burstein (2003) and Shermis et al. (2010) for an overview of the state of the art in this task). A major weakness of many existing scoring engines such as the Intelligent Essay Assessor<sup>TM</sup> (Landauer et al., 2003) is that they adopt a holistic scoring scheme, which summarizes the quality of an essay with a single score and thus provides very limited feedback to the writer. In particular, it is not clear which dimension of an essay (e.g., style, coherence, relevance) a score should be attributed to. Recent work addresses this problem by scoring a particular dimension of essay quality such as coherence (Miltsakaki and Kukich, 2004), technical errors, Relevance to Prompt (Higgins et al., 2004), and organization (Persing et al., 2010). Essay grading software that provides feedback along multiple dimensions of essay quality such as E-rater/Criterion (Attali and Burstein, 2006) has also begun to emerge.

Nevertheless, there is an essay scoring dimension for which few computational models have been developed — *thesis clarity*. Thesis clarity refers to how clearly an author explains the *thesis* of her essay, i.e., the position she argues for with respect to the topic on which the essay is written.<sup>1</sup> An essay with a high thesis clarity score presents its thesis in a way that is easy for the reader to understand, preferably but not necessarily directly, as in essays with explicit thesis sentences. It additionally contains no errors such as excessive misspellings that make it more difficult for the reader to understand the writer’s purpose.

Our goals in this paper are two-fold. First, we aim to develop a computational model for scoring the thesis clarity of student essays. Because there are many reasons why an essay may receive a low thesis clarity score, our second goal is to build a system for determining why an essay receives its score. We believe the feedback provided by this system will be more informative to a student than would a thesis clarity score alone, as it will help her understand which aspects of her writing need to be improved in order to better convey her thesis. To this end, we identify five common errors that impact thesis clarity, and our system’s purpose is to determine which of these errors occur in a given essay. We evaluate our thesis clarity scoring model and error identification system on a data set of 830 essays annotated with both thesis clarity scores and errors.

In sum, our contributions in this paper are three-fold. First, we develop a scoring model and error identification system for the thesis clarity dimension on student essays. Second, we use features explicitly designed for each of the identified error

---

<sup>1</sup>An essay’s thesis is the overall message of the *entire* essay. This concept is unbound from the the concept of thesis sentences, as even an essay that never explicitly states its thesis in any of its sentences may still have an overall message that can be inferred from the arguments it makes.

Topic	Languages	Essays
Most university degrees are theoretical and do not prepare students for the real world. They are therefore of very little value.	13	131
The prison system is outdated. No civilized society should punish its criminals: it should rehabilitate them.	11	80
In his novel <i>Animal Farm</i> , George Orwell wrote “All men are equal but some are more equal than others.” How true is this today?	10	64

Table 1: Some examples of writing topics.

types in order to train our scoring model, in contrast to many existing systems for other scoring dimensions, which use more general features developed without the concept of error classes. Third, we make our data set consisting of thesis clarity annotations of 830 essays publicly available in order to stimulate further research on this task. Since progress in thesis clarity modeling is hindered in part by the lack of a publicly annotated corpus, we believe that our data set will be a valuable resource to the NLP community.

## 2 Corpus Information

We use as our corpus the 4.5 million word International Corpus of Learner English (ICLE) (Granger et al., 2009), which consists of more than 6000 essays written by university undergraduates from 16 countries and 16 native languages who are learners of English as a Foreign Language. 91% of the ICLE texts are argumentative. We select a subset consisting of 830 argumentative essays from the ICLE to annotate and use for training and testing of our models of essay thesis clarity. Table 1 shows three of the thirteen topics selected for annotation. Fifteen native languages are represented in the set of essays selected for annotation.

## 3 Corpus Annotation

For each of the 830 argumentative essays, we ask two native English speakers to (1) score it along the thesis clarity dimension and (2) determine the subset of the five pre-defined errors that detracts from the clarity of its thesis.

**Scoring.** Annotators evaluate the clarity of each essay’s thesis using a numerical score from 1 to 4 at half-point increments (see Table 2 for a description of each score). This contrasts with previous work on essay scoring, where the corpus is

Score	Description of Thesis Clarity
4	essay presents a <b>very clear thesis</b> and requires little or no clarification
3	essay presents a <b>moderately clear thesis</b> but could benefit from some clarification
2	essay presents an <b>unclear thesis</b> and would greatly benefit from further clarification
1	essay presents <b>no thesis of any kind</b> and it is difficult to see what the thesis could be

Table 2: Descriptions of the meaning of scores.

annotated with a binary decision (i.e., *good* or *bad*) for a given scoring dimension (e.g., Higgins et al. (2004)). Hence, our annotation scheme not only provides a finer-grained distinction of thesis clarity (which can be important in practice), but also makes the prediction task more challenging.

To ensure consistency in annotation, we randomly select 100 essays to have graded by both annotators. Analysis of these essays reveals that, though annotators only exactly agree on the thesis clarity score of an essay 36% of the time, the scores they apply are within 0.5 points in 62% of essays and within 1.0 point in 85% of essays. Table 3 shows the number of essays that receive each of the seven scores for thesis clarity.

score	1.0	1.5	2.0	2.5	3.0	3.5	4.0
essays	4	9	52	78	168	202	317

Table 3: Distribution of thesis clarity scores.

**Error identification.** To identify what kinds of errors make an essay’s thesis unclear, we ask one of our annotators to write 1–4 sentence critiques of thesis clarity on 527 essays, and obtain our list of five common error classes by categorizing the things he found to criticize. We present our annotators with descriptions of these five error classes (see Table 4), and ask them to assign zero or more of the error types to each essay.

It is important to note that we ask our annotators to mark an essay with one of these errors only when the error makes the thesis less clear. So for example, an essay whose thesis is irrelevant to the prompt but is explicitly and otherwise clearly stated would not be marked as having a Relevance to Prompt error. If the irrelevant thesis is stated in such a way that its inapplicability to the prompt causes the reader to be confused about what the essay’s purpose is, however, then the essay would be assigned a Relevance to Prompt error.

To measure inter-annotator agreement on error identification, we ask both annotators to identify

<b>Id</b>	<b>Error</b>	<b>Description</b>
CP	<b>Confusing Phrasing</b>	The thesis is phrased oddly, making it hard to understand the writer’s point.
IPR	<b>Incomplete Prompt Response</b>	The thesis seems to leave some part of a multi-part prompt unaddressed.
R	<b>Relevance to Prompt</b>	The apparent thesis’s weak relation to the prompt causes confusion.
MD	<b>Missing Details</b>	The thesis leaves out important detail needed to understand the writer’s point.
WP	<b>Writer Position</b>	The thesis describes a position on the topic without making it clear that this is the position the writer supports.

Table 4: Descriptions of thesis clarity errors.

the errors in the same 100 essays that were doubly-annotated with thesis clarity scores. We then compute Cohen’s Kappa (Carletta, 1996) on each error from the two sets of annotations, obtaining an average Kappa value of 0.75, which indicates fair agreement. Table 5 shows the number of essays assigned to each of the five thesis clarity errors. As we can see, Confusing Phrasing, Incomplete Prompt Response, and Relevance to Prompt are the major error types.

error	CP	IPR	R	MD	WP
essays	152	123	142	47	39

Table 5: Distribution of thesis clarity errors.

**Relationship between clarity scores and error classes.** To determine the relationship between thesis clarity scores and the five error classes, we train a linear SVM regressor using the SVM<sup>light</sup> software package (Joachims, 1999) with the five error types as independent variables and the reduction in thesis clarity score due to errors as the dependent variable. More specifically, each training example consists of a target, which we set to the essay’s thesis clarity score minus 4.0, and six binary features, each of the first five representing the presence or absence of one of the five errors in the essay, and the sixth being a bias feature which we always set to 1. Representing the reduction in an essay’s thesis clarity score with its thesis clarity score minus 4.0 allows us to more easily interpret the error and bias weights of the trained system, as under this setup, each error’s weight should be a negative number reflecting how many points an essay loses due to the presence of that error. The bias feature allows for the possibility that an essay may lose points from its thesis clarity score for problems not accounted for in our five error classes. By setting this bias feature to 1, we tell our learner that an essay’s default score may be less than 4.0 because these other problems may lower the average score of otherwise perfect essays.

After training, we examined the weight parameters of the learned regressor and found that they

were all negative:  $-0.6$  for CP,  $-0.5998$  for IPR,  $-0.8992$  for R,  $-0.6$  for MD,  $-0.8$  for WP, and  $-0.1$  for the bias. These results are consistent with our intuition that each of the enumerated error classes has a negative impact on thesis clarity score. In particular, each has a demonstrable negative impact, costing essays an average of more than 0.59 points when it occurs. Moreover, this set of errors accounts for a large majority of all errors impacting thesis clarity because unenumerated errors cost essays an average of only one-tenth of one point on the four-point thesis clarity scale.

## 4 Error Classification

In this section, we describe in detail our system for identifying thesis clarity errors.

### 4.1 Model Training and Application

We recast the problem of identifying which thesis clarity errors apply to an essay as a multi-label classification problem, wherein each essay may be assigned zero or more of the five pre-defined error types. To solve this problem, we train five binary classifiers, one for each error type, using a one-versus-all scheme. So in the binary classification problem for identifying error  $e_i$ , we create one training instance from each essay in the training set, labeling the instance as positive if the essay has  $e_i$  as one of its labels, and negative otherwise. Each instance is represented by seven types of features, including two types of baseline features (Section 4.2) and five types of features we introduce for error identification (Section 4.3).

After creating training instances for error  $e_i$ , we train a binary classifier,  $b_i$ , for identifying which test essays contain error  $e_i$ . We use SVM<sup>light</sup> for classifier training with the regularization parameter,  $C$ , set to  $c_i$ . To improve classifier performance, we perform feature selection. While we employ seven types of features (see Sections 4.2 and 4.3), only the word n-gram features are subject to feature selection.<sup>2</sup> Specifically, we employ

<sup>2</sup>We do not apply feature selection to the remaining fea-

the top  $n_i$  n-gram features as selected according to information gain computed over the training data (see Yang and Pedersen (1997) for details). Finally, since each classifier assigns a real value to each test essay presented to it indicating its confidence that the essay should be assigned error  $e_i$ , we employ a classification threshold  $t_i$  to decide how high this real value must be in order for our system to conclude that an essay contains error  $e_i$ .

Using held-out validation data, we jointly tune the three parameters in the previous paragraph,  $c_i$ ,  $n_i$ , and  $t_i$ , to optimize the F-score achieved by  $b_i$  for error  $e_i$ .<sup>3</sup> However, an exact solution to this optimization problem is computationally expensive. Consequently, we find a local maximum by employing the simulated annealing algorithm (Kirkpatrick et al., 1983), altering one parameter at a time to optimize F-score by holding the remaining parameters fixed.

After training the classifiers, we use them to classify the test set essays. The test instances are created in the same way as the training instances.

## 4.2 Baseline Features

Our **Baseline** system for error classification employs two types of features. First, since labeling essays with thesis clarity errors can be viewed as a text categorization task, we employ lemmatized word unigram, bigram, and trigram features that occur in the essay that have not been removed by the feature selection parameter  $n_i$ . Because the essays vary greatly in length, we normalize each essay’s set of word features to unit length.

The second type of baseline features is based on random indexing (Kanerva et al., 2000). Random indexing is “an efficient, scalable and incremental alternative” (Sahlgren, 2005) to Latent Semantic Indexing (Deerwester et al., 1990; Landauer

ture types since each of them includes only a small number of overall features that are expected to be useful.

<sup>3</sup>For parameter tuning, we employ the following values.  $c_i$  may be assigned any of the values  $10^2$ ,  $10^3$ ,  $10^4$ ,  $10^5$ , or  $10^6$ .  $n_i$  may be assigned any of the values 3000, 4000, 5000, or ALL, where ALL means all features are used. For  $t_i$ , we split the range of classification values  $b_i$  returns for the test set into tenths.  $t_i$  may take the values 0.0, 0.1, 0.2, . . . , 1.0, and X, where 0.0 classifies all instances as negative, 0.1 classifies only instances  $b_i$  assigned values in the top tenth of the range as positive, and so on, and X is the default threshold, labeling essays as positive instances of  $e_i$  only if  $b_i$  returns for them a value greater than 0. It was necessary to assign  $t_i$  in this way because the range of values classifiers return varies greatly depending on which error type we are classifying and which other parameters we use. This method gives us reasonably fine-grained thresholds without having to try an unreasonably large number of values for  $t_i$ .

and Dutnais, 1997) which allows us to automatically generate a semantic similarity measure between any two words. We train our random indexing model on over 30 million words of the English Gigaword corpus (Parker et al., 2009) using the S-Space package (Jurgens and Stevens, 2010). We expect that features based on random indexing may be particularly useful for the Incomplete Prompt Response and Relevance to Prompt errors because they may help us find text related to the prompt even if some of its components have been rephrased (e.g., an essay may talk about “jail” rather than “prison”, which is mentioned in one of the prompts). For each essay, we therefore generate four random indexing features, one encoding the entire essay’s similarity to the prompt, another encoding the essay’s highest individual sentence’s similarity to the prompt, a third encoding the highest entire essay similarity to one of the prompt sentences, and finally one encoding the highest individual sentence similarity to an individual prompt sentence. Since random indexing does not provide a straightforward way to measure similarity between groups of words such as sentences or essays, we use Higgins and Burstein’s (2007) method to generate these features.

## 4.3 Novel Features

Next, we introduce five types of novel features.

**Spelling.** One problem we note when examining the information gain top-ranked features for the Confusing Phrasing error is that there are very few common confusing phrases that can contribute to this error. Errors of this type tend to be unique, and hence are not very useful for error classification (because we are not likely to see the same error in the training and test sets). We notice, however, that there are a few misspelled words at the top of the list. This makes sense because a thesis sentence containing excessive misspellings may be less clear to the reader. Even the most common spelling errors, however, tend to be rare. Furthermore, we ask our annotators to only annotate an error if it makes the thesis less clear. The mere presence of an awkward phrase or misspelling is not enough to justify the Confusing Phrasing label. Hence, we introduce a **misspelling** feature whose value is the number of spelling errors in an essay’s most-misspelled sentence.<sup>4</sup>

<sup>4</sup>We employ SCOWL (<http://wordlist.sourceforge.net/>) as our dictionary, assuming that a

**Keywords.** Improving the prediction of majority classes can greatly enhance our system’s overall performance. Hence, since we have introduced the misspelling feature to enhance our system’s performance on one of the more frequently occurring errors (Confusing Phrasing), it makes sense to introduce another type of feature to improve performance on the other two most frequent errors, Incomplete Prompt Response and Relevance to Prompt. For this reason, we introduce keyword features. To use this feature, we first examine each of the 13 essay prompts, splitting it into its component pieces. For our purposes, a component of a prompt is a prompt substring such that, if an essay does not address it, it may be assigned the Incomplete Prompt Response label. Then, for each component, we manually select the most important (primary) and second most important (secondary) words that it would be good for a writer to use to address the component. To give an example, the lemmatized version of the third component of the second essay in Table 1 is “it should rehabilitate they”. For this component we selected “rehabilitate” as a primary keyword and “society” as a secondary keyword. To compute one of our keyword features, we compute the random indexing similarity between the essay and each group of primary keywords taken from components of the essay’s prompt and assign the feature the lowest of these values. If this feature has a low value, that suggests that the essay may have an Incomplete Prompt Response error because the essay probably did not respond to the part of the prompt from which this value came. To compute another of the keyword features, we count the numbers of combined primary and secondary keywords the essay contains from each component of its prompt, and divide each number by the total number of primary and secondary features for that component. If the greatest of these fractions has a low value, that indicates the essay’s thesis might not be very Relevant to the Prompt.<sup>5</sup>

**Aggregated word n-gram features.** Other ways we could measure our system’s performance (such as macro F-score) would consider our system’s performance on the less frequent errors no less important than its performance on the

word that does not appear in the dictionary is misspelled.

<sup>5</sup>Space limitations preclude a complete listing of the keyword features. See our website at <http://www.hlt.utdallas.edu/~persingq/ICLE/> for the complete list.

most frequent errors. For this reason, it now makes sense for us to introduce a feature tailored to help our system do better at identifying the least-frequent error types, Missing Details and Writer Position, each of which occurs in fewer than 50 essays. To help with identification of these error classes, we introduce aggregated word n-gram features. While we mention in the previous section one of the reasons regular word n-gram features can be expected to help with these error classes, one of the problems with regular word n-gram features is that it is fairly infrequent for the exact same useful phrase to occur too frequently. Additionally, since there are numerous word n-grams, some infrequent ones may just by chance only occur in positive training set instances, causing the learner to think they indicate the positive class when they do not. To address these problems, for each of the five error classes  $e_i$ , we construct two Aggregated word features  $Aw+_i$  and  $Aw-_i$ . For each essay,  $Aw+_i$  counts the number of word n-grams we believe indicate that an essay is a positive example of  $e_i$ , and  $Aw-_i$  counts the number of word n-grams we believe indicate an essay is not an example of  $e_i$ .  $Aw+$  n-grams for the Missing Details error tend to include phrases like “there is something” or “this statement”, while  $Aw-$  ngrams are often words taken directly from an essay’s prompt. N-grams used for Writer Position’s  $Aw+$  tend to suggest the writer is distancing herself from whatever statement is being made such as “every person”, but n-grams for this error’s  $Aw-$  feature are difficult to find. Since  $Aw+_i$  and  $Aw-_i$  are so error specific, they are only included in an essay’s feature representation when it is presented to learner  $b_i$ . So while aggregated word n-grams introduce ten new features, each learner  $b_i$  only sees two of these ( $Aw+_i$  and  $Aw-_i$ ).

We construct the lists of word n-grams that are aggregated for use as the  $Aw+$  and  $Aw-$  feature values in the following way. For each error class  $e_i$ , we sort the list of all features occurring at least ten times in the training set by information gain. A human annotator then manually inspects the top thousand features in each of the five lists and sorts each list’s features into three categories. The first category for  $e_i$ ’s list consists of features that indicate an essay may be a positive instance. Each word n-gram from this list that occurs in an essay increases the essay’s  $Aw+_i$  value by one.

Similarly, any word n-gram sorted into the second category, which consists of features the annotator thinks indicate a negative instance of  $e_i$ , increases the essay’s  $Aw-$  value by one. The third category just contains all the features the annotator did not believe were useful enough to either class, and we make no further use of those features. For most error types, only about 12% of the top 1000 features get sorted into one of the first two categories.

**POS n-grams.** We might further improve our system’s performance on the Missing Details error type by introducing a feature that aggregates part-of-speech (POS) tag n-grams in the same way that the  $Aw$  features aggregate word n-gram features. For this reason, we include POS tag 1, 2, 3, and 4-grams in the set of features we sort in the previous paragraph. For each error  $e_i$ , we select POS tag n-grams from the top thousand features of the information gain sorted list to count toward the  $Ap+_i$  and  $Ap-_i$  aggregation features. We believe this kind of feature may help improve performance on Missing Details because the list of features aggregated to generate the  $Ap+_i$  feature’s value includes POS n-gram features like CC “NN” (scare quotes). This feature type may also help with Confusing Phrasing because the list of POS tag n-grams our annotator generated for its  $Ap+_i$  contains useful features like DT NNS VBZ VBN (e.g., “these signals has been”), which captures noun-verb disagreement.

**Semantic roles.** Our last aggregated feature is generated using FrameNet-style semantic role labels obtained using SEMAFOR (Das et al., 2010). For each sentence in our data set, SEMAFOR identifies each semantic frame occurring in the sentence as well as each frame element that participates in it. For example, a semantic frame may describe an event that occurs in a sentence, and the event’s frame elements may be the people or objects that participate in the event. For a more concrete example, consider the sentence “They said they do not believe that the prison system is outdated”. This sentence contains a Statement frame because a statement is made in it. One of the frame elements participating in the frame is the Speaker “they”. From this frame, we would extract a feature pairing the frame together with its frame element to get the feature “Statement-Speaker-they”. This feature indicates that the essay it occurs in might be a positive instance of the Writer Position error since it tells us the writer is

attributing some statement being made to someone else. Hence, this feature along with several others like “Awareness-Cognizer-we all” are useful when constructing the lists of frame features for Writer Position’s aggregated frame features  $Af+_i$  and  $Af-_i$ . Like every other aggregated feature,  $Af+_i$  and  $Af-_i$  are generated for every error  $e_i$ .

## 5 Score Prediction

Because essays containing thesis clarity errors tend to have lower thesis clarity scores than essays with fewer errors, we believe that thesis clarity scores can be predicted for essays by utilizing the same features we use for identifying thesis clarity errors. Because our score prediction system uses the same feature types we use for thesis error identification, each essay’s vector space representation remains unchanged. Only its label changes to one of the values in Table 2 in order to reflect its thesis clarity score. To make use of the fact that some pairs of scores are more similar than others (e.g., an essay with a score of 3.5 is more similar to an essay with a score of 4.0 than it is to one with a score of 1.0), we cast thesis clarity score prediction as a regression rather than classification task.

Treating thesis clarity score prediction as a regression problem removes our need for a classification threshold parameter like the one we use in the error identification problem, but if we use SVM<sup>light</sup>’s regression option, it does not remove the need for tuning a regularization parameter,  $C$ , or a feature selection parameter,  $n$ .<sup>6</sup> We jointly tune these two parameters to optimize performance on held-out validation data by performing an exhaustive search in the parameter space.<sup>7</sup>

After we select the features, construct the essay instances, train a regressor on training set essays, and tune parameters on validation set essays, we can use the regressor to obtain thesis clarity scores on test set essays.

<sup>6</sup>Before tuning the feature selection parameter, we have to sort the list of n-gram features occurring the training set. To enable the use of information gain as the sorting criterion, we treat each distinct score as its own class.

<sup>7</sup>The absence of the classification threshold parameter and the fact that we do not need to train multiple learners, one for each score, make it feasible for us to do two things. First, we explore a wider range of values for the two parameters: we allow  $C$  to take any value from  $10^0$ ,  $10^1$ ,  $10^2$ ,  $10^3$ ,  $10^4$ ,  $10^5$ ,  $10^6$ , or  $10^7$ , and we allow  $n$  to take any value from 1000, 2000, 3000, 4000, 5000, or ALL. Second, we exhaustively explore the space defined by these parameters in order to obtain an exact solution to the parameter optimization problem.

## 6 Evaluation

In this section, we evaluate our systems for error identification and scoring. All the results we report are obtained via five-fold cross-validation experiments. In each experiment, we use 3/5 of our labeled essays for model training, another 1/5 for parameter tuning, and the final 1/5 for testing.

### 6.1 Error Identification

**Evaluation metrics.** To evaluate our thesis clarity error type identification system, we compute precision, recall, micro F-score, and macro F-score, which are calculated as follows. Let  $tp_i$  be the number of test essays correctly labeled as positive by error  $e_i$ 's binary classifier  $b_i$ ;  $p_i$  be the total number of test essays labeled as positive by  $b_i$ ; and  $g_i$  be the total number of test essays that belong to  $e_i$  according to the gold standard. Then, the precision ( $P_i$ ), recall ( $R_i$ ), and F-score ( $F_i$ ) for  $b_i$  and the macro F-score ( $\hat{F}$ ) of the combined system for one test fold are calculated by

$$P_i = \frac{tp_i}{p_i}, R_i = \frac{tp_i}{g_i}, F_i = \frac{2P_iR_i}{P_i + R_i}, \hat{F} = \frac{\sum_i F_i}{5}.$$

However, the macro F-score calculation can be seen as giving too much weight to the less frequent errors. To avoid this problem, we also calculate for each system the micro precision, recall, and F-score (P, R, and F), where

$$P = \frac{\sum_i tp_i}{\sum_i p_i}, R = \frac{\sum_i tp_i}{\sum_i g_i}, F = \frac{2PR}{P + R}.$$

Since we perform five-fold cross-validation, each value we report for each of these measures is an average over its values for the five folds.<sup>8</sup>

**Results and discussion.** Results on error identification, expressed in terms of precision, recall, micro F-score, and macro F-score are shown in the first four columns of Table 6. Our **Baseline** system, which only uses word n-gram and random indexing features, seems to perform uniformly poorly across both micro and macro F-scores (F and  $\hat{F}$ ; see row 1). The per-class results<sup>9</sup> show that, since micro F-score places more weight on the correct identification of the most frequent errors, the system's micro F-score (31.1%) is fairly close to the average of the scores obtained on the three most frequent error classes, CP, IPR, and R,

<sup>8</sup>This averaging explains why the formula for F does not exactly hold in the Table 6 results.

<sup>9</sup>Per-class results are not shown due to space limitations.

System	Error Identification				Scoring		
	P	R	F	$\hat{F}$	S1	S2	S3
<b>B</b>	24.8	44.7	31.1	24.0	.658	.517	.403
<b>Bm</b>	24.2	44.2	31.2	25.3	.654	.515	.402
<b>Bmk</b>	29.2	44.2	34.9	26.7	.663	.490	<b>.369</b>
<b>Bmkw</b>	28.5	49.6	35.5	31.4	<b>.651</b>	.484	.374
<b>Bmkwp</b>	<b>34.2</b>	49.6	40.4	34.6	.671	<b>.483</b>	.377
<b>Bmkwpf</b>	33.6	<b>54.4</b>	<b>41.4</b>	<b>37.3</b>	.672	.486	.382

Table 6: Five-fold cross-validation results for the thesis clarity error identification and scoring.

and remains unaffected by very low F-scores on the two remaining infrequent classes.<sup>10</sup>

When we add the **misspelling** feature to the baseline, resulting in the system called **Bm** (row 2), the micro F-score sees a very small, insignificant improvement.<sup>11</sup> What is pleasantly surprising, however, is that, even though the misspelling features were developed for the Confusing Phrasing error type, they actually have more of a positive impact on Missing Details and Writer Position, bumping their individual error F-scores up by about 5 and 3 percent respectively. This suggests that spelling difficulties may be correlated with these other essay-writing difficulties, despite their apparent unrelatedness. This effect is strong enough to generate the small, though insignificant, gain in macro F-score shown in the table.

When we add **keyword** features to the system, micro F-score increases significantly by 3.7 points (row 3). The micro per-class results reveal that, as intended, keyword features improve Incomplete Prompt Response and Relevance to Prompt's F-scores reveals that they do by 6.4 and 9.2 percentage points respectively. The macro F-scores reveal this too, though the macro F-score gains are 3.2 points and 11.5 points respectively. The macro F-score of the overall system would likely have improved more than shown in the table if the addition of keyword features did not simultaneously reduce Missing Details's score by several points.

While we hoped that adding aggregated word n-gram features to the system (row 4) would be able to improve performance on Confusing Phrasing due to the presence of phrases such as "in university be" in the error's  $Aw+_i$  list, there turned out to be few such common phrases in the data set,

<sup>10</sup>Since parameters for optimizing micro F-score and macro F-score are selected independently, the per-class F-scores associated with micro F-score are different than those used for calculating macro F-score. Hence, when we discuss per-class changes influencing micro F-score, we refer to the former set, and otherwise we refer to the latter set.

<sup>11</sup>All significance tests are paired  $t$ -tests, with  $p < 0.05$ .

so performance on this class remains mostly unchanged. This feature type does, however, result in major improvements to micro and macro performance on Missing Details and Writer Position, the other two classes this feature was designed to help. Indeed, the micro F-score versions of Missing Details and Writer Position improve by 15.3 and 10.8 percentage points respectively. Since these are minority classes, however, the large improvements result in only a small, insignificant improvement in the overall system’s micro F-score. The macro F-score results for these classes, however, improve by 6.5% and 17.6% respectively, giving us a nearly 5-point, statistically significant bump in macro F-score after we add this feature.

Confusing Phrasing has up to now stubbornly resisted any improvement, even when we added features explicitly designed to help our system do better on this error type. When we add aggregated part of speech n-gram features on top of the previous system, that changes dramatically. Adding these features makes both our system’s F-scores on Confusing Phrasing shoot up almost 8%, resulting in a significant, nearly 4.9% improvement in overall micro F-score and a more modest but insignificant 3.2% improvement in macro F-score (row 5). The micro F-score improvement can also be partly attributed to a four point improvement in Incomplete Prompt Response’s micro F-score. The 13.7% macro F-score improvement of the Missing Details error plays a larger role in the overall system’s macro F-score improvement than Confusing Phrasing’s improvement, however.

The improvement we see in micro F-score when we add aggregated frame features (row 6) can be attributed almost solely to improvements in classification of the minority classes. This is surprising because, as we mentioned before, minority classes tend to have a much smaller impact on overall micro F-score. Furthermore, the overall micro F-score improvement occurs despite declines in the performances on two of the majority class errors. Missing Details and Writer Position’s micro F-score performances increase by 19.1% and 13.4%. The latter is surprising only because of the magnitude of its improvement, as this feature type was explicitly intended to improve its performance. We did not expect this aggregated feature type to be especially useful for Missing Details error identification because very few of these types of features occur in its  $Af+i$  list, and there are

none in its  $Af-i$  list. The few that are in the former list, however, occur fairly often and look like fairly good indicators of this error (both the examples “Event-Event-it” and “Categorization-Item-that” occur in the positive list, and both do seem vague, indicating more details are to be desired).

Overall, this system improves our baseline’s macro F-score performance significantly by 13.3% and its micro F-score performance significantly by 10.3%. As we progressed, adding each new feature type to the baseline system, there was no definite and consistent pattern to how the precisions and recalls changed in order to produce the universal increases in the F-scores that we observed for each new system. Both just tended to jerkily progress upward as new feature types were added. This confirms our intuition about these features – namely that they do not all uniformly improve our performance in the same way. Some aim to improve precision by telling us when essays are less likely to be positive instances of an error class, such as any of the  $Aw-i$ ,  $Ap-i$ , or  $Af-i$  features, and others aim to tell us when an essay is more likely to be a positive instance of an error.

## 6.2 Scoring

**Scoring metrics.** We design three evaluation metrics to measure the error of our thesis clarity scoring system. The  $S1$  metric measures the frequency at which a system predicts the wrong score out of the seven possible scores. Hence, a system that predicts the right score only 25% of the time would receive an  $S1$  score of 0.75.

The  $S2$  metric measures the average distance between the system’s score and the actual score. This metric reflects the idea that a system that estimates scores close to the annotator-assigned scores should be preferred over a system whose estimations are further off, even if both systems estimate the correct score at the same frequency.

Finally, the  $S3$  metric measures the average square of the distance between a system’s thesis clarity score estimations and the annotator-assigned scores. The intuition behind this metric is that not only should we prefer a system whose estimations are close to the annotator scores, but we should also prefer one whose estimations are not too frequently very far away from the annotator scores. These three scores are given by:

$$\frac{1}{N} \sum_{A_j \neq E'_j} 1, \quad \frac{1}{N} \sum_{i=1}^N |A_j - E_j|, \quad \frac{1}{N} \sum_{i=1}^N (A_j - E_j)^2$$

where  $A_j$ ,  $E_j$ , and  $E'_j$  are the annotator assigned, system estimated, and rounded system estimated scores<sup>12</sup> respectively for essay  $j$ , and  $N$  is the number of essays.

**Results and discussion.** Results on scoring are shown in the last three columns of Table 6. We see that the thesis clarity score predicting variation of the **Baseline** system, which employs as features only word n-grams and random indexing features, predicts the wrong score 65.8% of the time. Its predicted score is on average 0.517 points off of the actual score, and the average squared distance between the predicted and actual scores is 0.403.

We observed earlier that a high number of misspellings may be positively correlated with one or more unrelated errors. Adding the **misspelling** feature to the scoring systems, however, only yields minor, insignificant improvements to their performances under the three scoring metrics.

While adding **keyword** features on top of this system does not improve the frequency with which the right score is predicted, it both tends to move the predictions closer to the actual thesis clarity score value (as evidenced by the significant improvement in  $S2$ ) and ensures that predicted scores will not too often stray too far from the actual value (as evidenced by the significant improvement in  $S3$ ). Overall, the scoring model employing the **Bmk** feature set performs significantly better than the **Baseline** scoring model with respect to two out of three scoring metrics.

The only remaining feature type whose addition yields a significant performance improvement is the aggregated **word** feature type, which improves system **Bmk**'s  $S2$  score significantly while having an insignificant impact on the other  $S$  metrics.

Neither of the remaining aggregative features yields any significant improvements in performance. This is a surprising finding since, up until we introduced aggregated **part-of-speech** tag n-gram features into our regressor, each additional feature that helped with error classification made at least a small but positive contribution to at least two out of the three  $S$  scores. These aggregative features, which proved to be very powerful when assigning error labels, are not as useful for thesis

<sup>12</sup>Since our regressor assigns each essay a real value rather than an actual valid thesis clarity score, it would be difficult to obtain a reasonable  $S1$  score without rounding the system estimated score to one of the possible values. For that reason, we round the estimated score to the nearest of the seven scores the human annotators were permitted to assign (1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0) only when calculating  $S1$ .

Gold	S1 (Bmkw)			S2 (Bmkwp)			S3 (Bmk)		
	.25	.50	.75	.25	.50	.75	.25	.50	.75
1.0	3.5	3.5	3.5	3.0	3.2	3.5	3.1	3.2	3.3
1.5	2.5	3.0	3.0	2.8	3.1	3.2	2.6	3.0	3.2
2.0	3.0	3.0	3.5	3.0	3.2	3.5	3.0	3.1	3.4
2.5	3.0	3.5	3.5	3.0	3.3	3.6	3.0	3.3	3.5
3.0	3.0	3.5	3.5	3.1	3.4	3.5	3.1	3.3	3.5
3.5	3.5	3.5	4.0	3.2	3.4	3.6	3.2	3.4	3.5
4.0	3.5	3.5	4.0	3.4	3.6	3.8	3.4	3.5	3.7

Table 7: Regressor scores for top three systems.

clarity scoring.

To more closely examine the behavior of the best scoring systems, in Table 7 we chart the distributions of scores they predict for each gold standard score. As an example of how to read this table, consider the number 2.8 appearing in row 1.5 in the .25 column of the  $S2$  (**Bmkwp**) region. This means that 25% of the time, when system **Bmkwp** (which obtains the best  $S2$  score) is presented with a test essay having a gold standard score of 1.5, it predicts that the essay has a score less than or equal to 2.8 for the  $S2$  metric.

From this table, we see that each of the best systems has a strong bias toward predicting more frequent scores as there are no numbers less than 3.0 in the 50% columns, and about 82.8% of all essays have gold standard scores of 3.0 or above. Nevertheless, no system relies entirely on bias, as evidenced by the fact that each column in the table has a tendency for its scores to ascend as the gold standard score increases, implying that the systems have some success at predicting lower scores for essays with lower gold standard scores.

Finally, we note that the difference in error weighting between the  $S2$  and  $S3$  scoring metrics appears to be having its desired effect, as there is a strong tendency for each entry in the  $S3$  subtable to be less than or equal to its corresponding entry in the  $S2$  subtable due to the greater penalty the  $S3$  metric imposes for predictions that are very far away from the gold standard scores.

## 7 Conclusion

We examined the problem of modeling thesis clarity errors and scoring in student essays. In addition to developing these models, we proposed novel features for use in our thesis clarity error model and employed these features, each of which was explicitly designed for one or more of the error types, to train our scoring model. We make our thesis clarity annotations publicly available in order to stimulate further research on this task.

## Acknowledgments

We thank the three anonymous reviewers for their detailed and insightful comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-1147644 and IIS-1219142. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of NSF.

## References

- Yigal Attali and Jill Burstein. 2006. Automated essay scoring with E-rater v.2.0. *Journal of Technology, Learning, and Assessment*, 4(3).
- Jean Carletta. 1996. Assessing agreement on classification tasks: The Kappa statistic. *Computational Linguistics*, 22(2):249–254.
- Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 948–956.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.
- Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. *International Corpus of Learner English (Version 2)*. Presses universitaires de Louvain.
- Derrick Higgins and Jill Burstein. 2007. Sentence similarity measures for essay coherence. In *Proceedings of the 7th International Workshop on Computational Semantics*.
- Derrick Higgins, Jill Burstein, Daniel Marcu, and Claudia Gentile. 2004. Evaluating multiple aspects of coherence in student essays. In *Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 185–192.
- Thorsten Joachims. 1999. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, Chapter 11, pages 169–184. MIT Press, Cambridge, MA.
- David Jurgens and Keith Stevens. 2010. The S-Space package: An open source package for word space models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 30–35.
- Pentti Kanerva, Jan Kristoferson, and Anders Holst. 2000. Random indexing of text samples for Latent Semantic Analysis. In *Proceedings the 22nd Annual Conference of the Cognitive Science Society*, pages 103–106.
- Scott Kirkpatrick, C. D. Gelatt, and Mario P. Vecchi. 1983. Optimization by simulated annealing. *Science*, 220(4598):671–680.
- Thomas K Landauer and Susan T. Dumais. 1997. A solution to Plato’s problem: The Latent Semantic Analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240.
- Thomas K. Landauer, Darrell Laham, and Peter W. Foltz. 2003. Automated scoring and annotation of essays with the Intelligent Essay Assessor™. In *Automated Essay Scoring: A Cross-Disciplinary Perspective*, pages 87–112. Lawrence Erlbaum Associates, Inc., Mahwah, NJ.
- Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25–55.
- Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2009. *English Gigaword Fourth Edition*. Linguistic Data Consortium, Philadelphia.
- Isaac Persing, Alan Davis, and Vincent Ng. 2010. Modeling organization in student essays. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 229–239.
- Magnus Sahlgren. 2005. An introduction to random indexing. In *Proceedings of the Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering*.
- Mark D. Shermis and Jill C. Burstein. 2003. *Automated Essay Scoring: A Cross-Disciplinary Perspective*. Lawrence Erlbaum Associates, Inc., Mahwah, NJ.
- Mark D. Shermis, Jill Burstein, Derrick Higgins, and Klaus Zechner. 2010. Automated essay scoring: Writing assessment and instruction. In *International Encyclopedia of Education (3rd edition)*. Elsevier, Oxford, UK.
- Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420.

# Translating Italian connectives into Italian Sign Language

**Camillo Lugaresi**

University of Illinois at Chicago  
Politecnico di Milano  
clugar2@uic.edu

**Barbara Di Eugenio**

Department of Computer Science  
University of Illinois at Chicago  
bdieugen@uic.edu

## Abstract

We present a corpus analysis of how Italian connectives are translated into LIS, the Italian Sign Language. Since corpus resources are scarce, we propose an alignment method between the syntactic trees of the Italian sentence and of its LIS translation. This method, and clustering applied to its outputs, highlight the different ways a connective can be rendered in LIS: with a corresponding sign, by affecting the location or shape of other signs, or being omitted altogether. We translate these findings into a computational model that will be integrated into the pipeline of an existing Italian-LIS rendering system. Initial experiments to learn the four possible translations with Decision Trees give promising results.

## 1 Introduction

Automatic translation between a spoken language and a signed language gives rise to some of the same difficulties as translation between spoken languages, but adds unique challenges of its own. Contrary to what one might expect, sign languages are not artificial languages, but natural languages that spontaneously arose within deaf communities; although they are typically named after the region where they are used, they are not derived from the local spoken language and tend to bear no similarity to it. Therefore, translation from any spoken language into the signed language of that specific region is at least as complicated as between any pairs of unrelated languages.

The problem of automatic translation is compounded by the fact that the amount of computational resources to draw on is much smaller than is typical for major spoken languages. Moreover, the fact that sign languages employ a different

transmission modality (gestures and expressions instead of sounds) means that existing writing systems are not easily adaptable to them. The resulting lack of a shared written form does nothing to improve the availability of sign language corpora; bilingual corpora, which are of particular importance to a translation system, are especially rare. In fact, various projects around the world are trying to ameliorate this sad state of affairs for specific Sign Languages (Lu and Huenerfauth, 2010; Braffort et al., 2010; Morrissey et al., 2010).

In this paper, we describe the work we performed as concerns the translation of connectives from the Italian language into LIS, the Italian Sign Language (*Lingua Italiana dei Segni*). Because the communities of signers in Italy are relatively small and fragmented, and the language has a relatively short history, there is far less existing research and material to draw on than for, say, ASL (American Sign Language) or BSL (British Sign Language).

Our work was undertaken within the purview of the ATLAS project (Bertoldi et al., 2010; Lombardo et al., 2010; Lombardo et al., 2011; Prinetto et al., 2011; Mazzei, 2012; Ahmad et al., 2012), which developed a full pipeline for translating Italian into LIS. ATLAS is part of a recent crop of projects devoted to developing automatic translation from language  $L$  spoken in geographic area  $G$  into the sign language spoken in  $G$  (Dreuw et al., 2010; López-Ludeña et al., 2011; Almohimeed et al., 2011; Lu and Huenerfauth, 2012). Input is taken in the form of written Italian text, parsed, and converted into a semantic representation of its contents; from this semantic representation, LIS output is produced, using a custom serialization format called AEWLIS (which we will describe later). This representation is then augmented with space positioning information, and fed into a final renderer component that performs the signs using a virtual actor. ATLAS focused on a limited domain for which a bilingual Italian/LIS cor-

pus was available: weather forecasts, for which the Italian public broadcasting corporation (RAI) had long been producing special broadcasts with a signed translation. This yielded a corpus of 376 LIS sentences with corresponding Italian text: this corpus, converted into AEWLIS format, was the main data source for the project. Still, it is a very small corpus, hence the main project shied away from statistical NLP techniques, relying instead on rule-based approaches developed with the help of a native Italian/LIS bilingual speaker; a similar approach is taken e.g. in (Almohimeed et al., 2011) for Arabic.

### 1.1 Why connectives?

The main semantic-bearing elements of an Italian sentence, such as nouns or verbs, typically have a LIS sign as their direct translation. We focus on a different class of elements, comprising conjunctions and prepositions, but also some adverbs and prepositional phrases; collectively, we refer to them as connectives. Since they are mainly structural elements, they are more heavily affected by differences in the syntax and grammar of Italian and LIS (and, presumably, in those of any spoken language and the “corresponding” SL). Specifically, as we will see later, some connectives are translated with a sign, some connectives are dropped, whereas others affect the positioning of other signs, or just their syntactic proximity.

It should be noted that our usage of the term “connectives” is somewhat unorthodox. For example, while prepositions can be seen as connectives (Ferrari, 2008), only a few adverbs can work as connectives. From the Italian Treebank, we extracted all words or phrases that belonged to a syntactic category that can be a connective (conjunction, preposition, adverb or prepositional phrase). We then found that we could better serve the needs of ATLAS by running our analysis on the entire resulting list, without filtering it by eliminating the entries that are not actual connectives. In fact, semantic differences re-emerge through our analysis: e.g., the temporal adverbs “domani” and “dopodomani” are nearly always preserved, as they do carry key information (especially for weather forecasting) and are not structural elements.

In performing our analysis, we pursued a different path from the main project, relying entirely on the bilingual corpus. Although the use of sta-

tistical techniques was hampered by the small size of the corpus, at the same time it presented an interesting opportunity to attack the problem from a different angle. In this paper we describe how we uncovered the translation distributions of the different connectives from Italian to LIS via tree alignment.

## 2 Corpus Analysis

The corpus consists of 40 weather forecasts in Italian and LIS. The Italian spoken utterance and LIS signing were transcribed from the original videos – one example of an Italian sentence and its LIS equivalent are shown in Figure 1. An English word-by-word translation is provided for the Italian sentence, followed by a more fluent translation; the LIS glosses are literally translated. Note that as concerns LIS, this simply includes the gloss for the corresponding sign. The 40 weather forecast comprise 374 Italian sentences and 376 LIS sentences, stored in 372 AEWLIS files. In most cases, a file corresponds to one Italian sentence and one corresponding LIS sentences; however, there are 4 files where an Italian sentence is split into two LIS sentences, and 2 files where two Italian sentences are merged into one LIS sentence.

AEWLIS is an XML-based format (see Figure 2) which represents each sign in the LIS sentence as an element, in the order in which they occur in the sentence. A sign’s lemma is represented by the Italian word with the same meaning, always written in uppercase, and with its part of speech (*tipoAG* in Figure 2); there are also IDs referencing the lemma’s position in a few dictionaries, but these are not always present. The AEWLIS file also stores several additional attributes, such as: a parent reference that represents the syntax of the LIS sentence; the syntactic role “played” by the sign in the LIS sentence; the facial expression accompanying the gesture; the location in the signing space (which may be an absolute location or a reference to a previous sign’s: compare *HR* (High Right) and *atLemma* in Figure 2). These attributes are stored as elements grouped by type, and reference the corresponding sign element by its ordinal position in the sentence. The additional attributes are not always available: morphological variations are annotated only when they differ from an assumed standard form of the sign, while the syntactic structure was annotated for only 89 sentences.

- (1) (Ita.) Anche sulla Sardegna qualche annuvolamento pomeridiano, possibilità di qualche breve scroscio di pioggia,  
 Also on Sardinia a few cloud covers afternoon[adj], chance of a few brief downpour of rain,  
 ma tendenza poi a schiarite.  
 but trend then towards sunny spells.  
 “Also on Sardinia skies will become overcast in the afternoon, chance of a few brief downpours of rain, but then a trend towards a mix of sun and clouds”.
- (2) (LIS) POMERIGGIO SARDEGNA AREA NUVOLA PURE ACQUAZZONE POTERE MA POI NUVOLA  
 Afternoon Sardinia area cloud also downpour can[modal] but then cloud  
 DIMINUIRE  
 decrease

Figure 1: Italian sentence and its LIS translation

```
<Lemma>
<NuovoLemma lemma="POMERIGGIO" tipoAG="NOME" ... endTime="2.247" idSign="" />
<NuovoLemma lemma="sardegna" tipoAG="NOME_PROPRIO" ... endTime="2.795" idSign="2687" />
<NuovoLemma lemma="area" tipoAG="NOME" ... endTime="4.08" idSign="2642" />
<NuovoLemma lemma="nuvola" tipoAG="NOME" ... endTime="5.486" idSign="2667" />
<NuovoLemma lemma="pure" tipoAG="AVVERBIO" ... endTime="6.504" idSign="2681" />
...
</Lemma><SentenceAttribute>
<Parent>
  <Timestamp time="1" value="ID:3" /> <Timestamp time="2" value="ID:2" />
  <Timestamp time="3" value="ID:3" /> <Timestamp time="4" value="root_1" />
  <Timestamp time="5" value="ID:3" /> ...
</Parent>
...
<Sign_Spatial_Location>
  <Timestamp time="1" value="" /> <Timestamp time="2" value="HR" />
  <Timestamp time="3" value="HL" /> <Timestamp time="4" value="atLemma(ID:2, Distant)" />
  <Timestamp time="5" value="" /> ...
</Sign_Spatial_Location>
...
<Facial>
  <Timestamp time="1" value="" /> <Timestamp time="2" value="eye brows:raise" />
  <Timestamp time="3" value="eye brows:raise" /> <Timestamp time="4" value="eye brows:-lwr" />
  <Timestamp time="5" value="" /> ...
</Facial>
</SentenceAttribute>
```

Figure 2: Example excerpt from an AEWLIS file

## 2.1 Distributional statistics for connectives

The list of Italian connectives we considered was extracted from the Italian Treebank developed at the Institute for Computational Linguistics in Pisa, Italy (Montemagni et al., 2003) by searching for conjunctions, prepositions and adverbs. This yielded a total of 777 potential connectives. Of those, only 104 occur in our corpus. A simple count of the occurrences of connectives in the Italian and LIS versions of the corpus yields the following results:

- (a) 78 connectives (2068 occurrences total) only occur in the Italian version, for example *ALMENO* (at least), *CON* (with), *INFATTI* (indeed), *PER* (for).
- (b) 8 connectives (67 occurrences total) only occur in the LIS version, for example *CIRCA* (about), *as in* “Here I am”, *PURE* (also, additionally).
- (c) 25 connectives (925 occurrences total) occur in both versions.

For the third category, we have computed the ratio of the number of occurrences in Italian over the number of occurrences in LIS; the ratios are plotted in logarithmic scale in Figure 3. 0 on the scale corresponds to an ITA/LIS ratio equal to 1; positive numbers indicate that there are more occurrences in ITA, negative numbers that there are more occurrences in LIS. We can recognize three clusters by ratio:

- (c1) 9 connectives occurring in both languages, but mainly in Italian, for example *POCO* (a little), *PIÙ* (more), *SE* (if), *QUINDI* (hence).
- (c2) 13 connectives occurring in both languages with similar frequency, for example *SOLO* (only), *POI* (then), *O* (or), *MA* (but).
- (c3) 3 connectives occurring in both languages, but mainly in LIS: *MENO* (less), *ADESSO* (now), *INVECE* (instead).

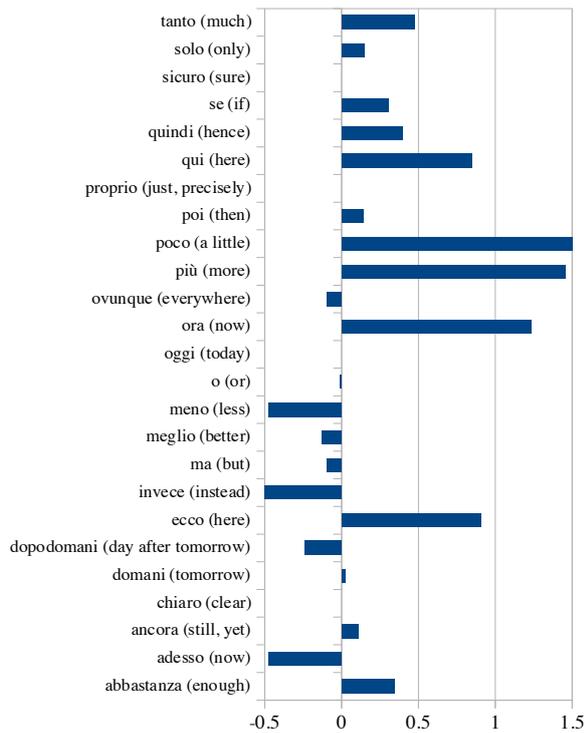


Figure 3: Ratio of ITA/LIS occurrences in logarithmic scale.

### 3 The effect of the Italian connectives on the LIS translation

From this basic frequency analysis we can already notice that a large number of connectives only appear in Italian, or have far more occurrences in Italian than in LIS. This is unsurprising, considering that LIS sentences tend to be shorter than Italian sentences in terms of number of signs/words (a fact which probably correlates with the increased energy and time requirements intrinsic into articulating a message using one's arms rather than one's tongue). However, our goal is to predict when a connective should be dropped and when it should be preserved. Furthermore, even if the connective does not appear in the LIS sentence as a directly corresponding sign, that does not mean that its presence in the Italian sentence has no effect on the translation. We hypothesize four different possible realizations for a connective in the Italian sentence:

- the connective word or phrase may map to a corresponding connective sign;
- the connective is not present as a sign, but may affect the morphology of the signs which translate words syntactically adjacent to the

connective;

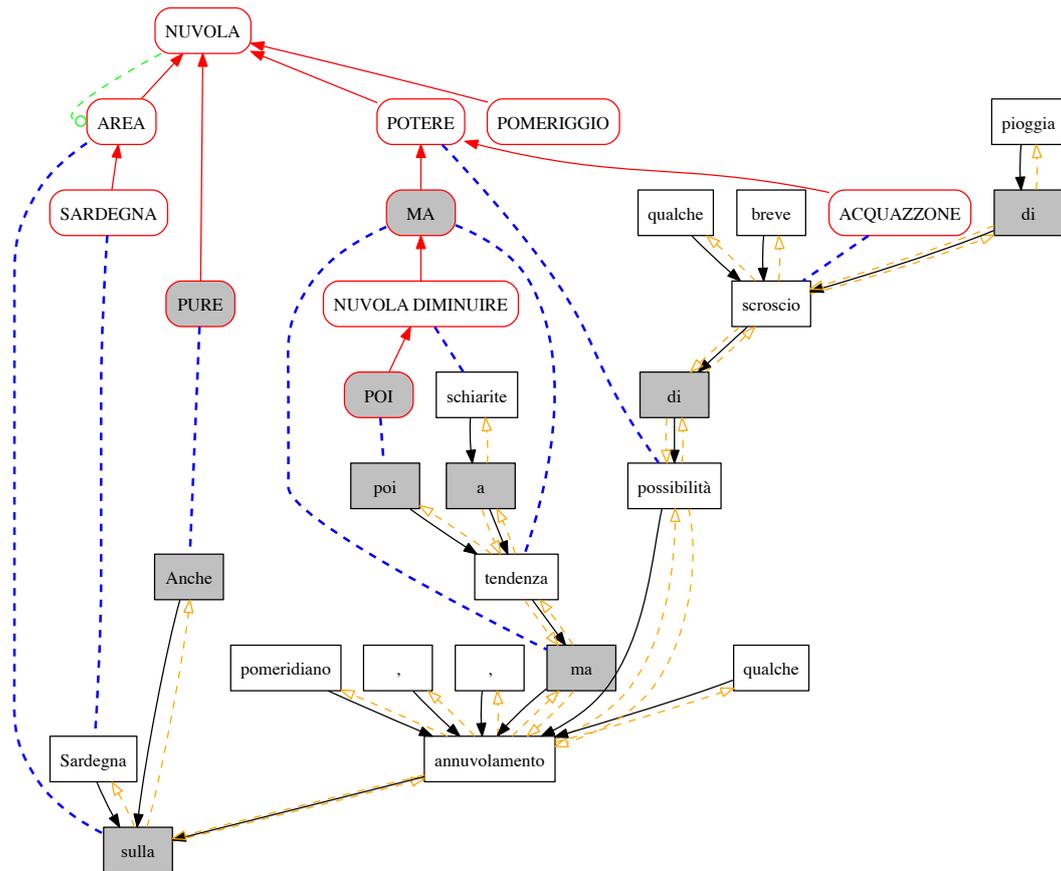
- the connective is not present as a sign, but its presence may be reflected by the fact that words connected by it map to signs which are close to each other in the LIS syntax tree;
- the connective is dropped altogether.

The second hypothesis deserves some explanation. The earliest treatments of LIS assumed that each sign (lemma) could be treated as invariant. Attempts to represent LIS in writing simply replaced each sign with a chosen Italian word (or phrase, if necessary) with the same meaning. Although this is still a useful way of representing the basic lemma, more recent studies have noted that LIS signs can undergo significant morphological variations which are lost under such a scheme. The AEWLIS format, in fact, was designed to preserve them.

Of course, morphological variations in LIS are not phonetic, like in a spoken language, but gestural (Volterra, 1987; Romeo, 1991). For example, the location in which a gesture is performed may be varied, or its speed, or the facial expressions that accompany it (Geraci et al., 2008). One particularly interesting axis of morphology is the positioning of the gesture in the signing space in front of the signer. This space is implicitly divided into a grid with a few different positions from left to right and from top to bottom (see HR – High Right, and LH – High Left, in Figure 2). Two or more signs can then be placed in different positions in this virtual space, and by performing other signs in the same positions the signer can express a backreference to the previously established entity at that location. One can even have a movement verb where the starting and ending positions of the gesture are positioned independently to indicate the source and destination of the movement. In other words, these morphological variations can perform a similar function to gender and number agreement in Italian backreferences, but they can also assume roles that in Italian would be performed by prepositions, which are connectives. In fact, as we will see later on, Italian prepositions are never translated as signs, but are often associated with morphological variations on related signs.

#### 3.1 Tree Alignment

Two of our four translation hypotheses involve a notion of distance on the syntax tree, and a no-



19\_f08\_2011-06-08\_10\_04\_10.xml  
 Anche sulla Sardegna qualche annuvolamento pomeridiano, possibilità di qualche breve scroscio di pioggia, ma tendenza poi a schiarite pomeriggio Sardegna area nuvola pure acquazzone potere ma poi nuvola diminuire

Figure 4: Example of integrated syntax trees.

tion of signs corresponding to words. Therefore, it is not sufficient to consider the LIS sentence and the Italian sentence separately. Instead, their syntax trees must be reconstructed and aligned. Tree alignment in a variety of forms has been extensively used in machine translation systems (Gildea, 2003; Eisner, 2003; May and Knight, 2007). As far as we know, we are the first to attempt the usage of tree alignment to aid in the translation between a spoken and a sign language, partly because corpora that include syntactic trees for sign language sentences hardly exist. (López-Ludeña et al., 2011) does use alignment techniques for translation from Spanish to Spanish Sign Language (SSL), but it is limited to alignment between words or phrases in Spanish, and glosses or sequences of glosses in SSL.

We have developed a pipeline that takes in input the corpus files, parses the Italian sentence with

an existing parser, and retrieves / builds a parse tree for the LIS sentence. The two trees are then aligned by exploiting the word/sign alignment. A sample output is shown in Figure 4.

**Italian sentence parsing.** Since the corpus contains the Italian sentences in plain, unstructured text form, they need to be parsed. We used the DeSR parser, a dependency parser pre-trained on a very large Italian corpus (Attardi et al., 2007; Ciarrita and Attardi, 2011). This parser produced the syntax trees and POS tagging that we used for the Italian part of the corpus.

**LIS syntax tree.** One of the attributes allowed by AEWLIS is “parent”, which points a sign to its parent in the syntax tree, or marks it as a root (see Figure 2). These hand-built syntax trees are available in roughly 1/4 of the AEWLIS files. Because the size of our corpus is already limited, and be-

cause no tools are available to generate LIS syntax trees, for the remaining 3/4 of the corpus we fell back on a simple linear tree where each sign is connected to its predecessor. This solution at least maintains locality in most cases.

**Word Alignment.** Having obtained syntax trees for the two sentences, we then needed to align them. For this purpose we used the Berkeley Word Aligner (BWA) <sup>1</sup> (Denero, 2007), a general tool for aligning sentences in bilingual corpora. BWA takes as input a series of matching sentences in two different languages, trains multiple unsupervised alignment models, and selects the optimal result using a testing set of manual alignments. The output is a list of aligned word indices for each input sentence pair. On our data set, BWA performance is as follows: Precision = 0.736364, Recall = 0.704348, AER = 0.280000.

**Integration.** The result is an integrated syntax tree representation of the Italian and LIS versions of the sentence, with arcs bridging aligned word/sign pairs. Since some connectives consist of multi-word phrases, the word nodes which are part of one are merged into a super-node that inherits all connections to other nodes. Figure 4 shows the end result for the Italian and LIS sentences in Figure 1 (the two sentences are repeated for convenience at the bottom of Figure 4). The rectangular boxes are words in the Italian sentence, while the rounded boxes are signs in the LIS sentence. The Italian tree has its root(s) at the bottom, while the LIS tree has its root(s) at the top. Solid arrows point from children to parent nodes in the syntax tree. Gray-shaded boxes represent connectives (words or signs, as indicated by the border of the box). Bold dashed lines show word alignment. Edges with round heads show relationships where a sign has a location attribute referencing another sign. Arrows with an empty triangular head trace the paths described in the next section.

### 3.2 Subtree alignment and path processing

At this point individual words are aligned, but that is not sufficient. Our hypotheses on the effect of connectives on translation requires us to align a tree fragment surrounding the Italian connective with the corresponding tree fragment on the LIS

side - where the connective may be missing. In effect, since we have hypothesized that the presence of a connective can affect the translation of the two subtrees that it connects, we would like to be able to align each of those subtrees to its translation. However, given the differences between the two languages, it is not easy to give a clear definition of this mapping - let alone to compute it.

Instead, we can take a step back to word-level alignment. We make the observation that, if two words belong to two different subtrees linked by a connective, so that the path between the two words goes through the connective, then the frontier between the LIS counterparts of those two subtrees should also lie along the path between the signs aligned with those two words. If the connective is preserved in translation as a sign, we should expect to find it along that path; if it is not, its effect should still be seen along that path, either in the form of morphological variations to the signs along the path, or in the shortness of the path itself.

The first step, then, is to split the Italian syntax tree by removing the connective. This yields one subtree containing the connective's parent node, if any, and one subtree for each of the connective's children, if any. The parent subtree typically contains most of the rest of the sentence, so only the direct ancestors of the connective are considered. Then, each pair of words belonging to different subtrees is linked by a path that goes through the connective in the original tree. Of these words, we select the ones that have aligned signs, and then we compute the path between each pair of signs aligned to words belonging to different subtrees. This gives us a set of paths to consider in the LIS syntax tree.

For example, let us consider the connective “di” between “possibilità” and “scroscio” in Figure 4.

- This node connects two subtrees: a child subtree containing “qualche, breve, scroscio, di, pioggia”, and a parent subtree containing the rest of the sentence.
- From each subtree, a set of paths is generated: all paths extending from the connective to the leaves of the child subtree (for example “scroscio, qualche” or “scroscio, di, pioggia”), and the path of direct ancestors in the parent tree (“sulla, annuolamento, possibilità”).
- Iterate through the cartesian product of each

<sup>1</sup><http://code.google.com/p/berkeleyaligner/>

Table 1: Translation candidates for connectives with more than 10 occurrences

Connective	ITA Occurrences	Sign	Location	Close	Missing
domani	71	67 (94.37%)	1 (1.41%)	2 (2.82%)	4 (5.63%)
dopodomani	15	14 (93.33%)	0 (0.00%)	0 (0.00%)	1 (6.67%)
mentre	28	26 (92.86%)	5 (17.86%)	0 (0.00%)	1 (3.57%)
o	37	37 (100.00%)	2 (5.41%)	6 (16.22%)	0 (0.00%)
però	10	9 (90.00%)	1 (10.00%)	1 (10.00%)	1 (10.00%)
ancora	72	44 (61.11%)	1 (1.39%)	3 (4.17%)	25 (34.72%)
invece	17	9 (52.94%)	1 (5.88%)	2 (11.76%)	6 (35.29%)
ma	51	29 (56.86%)	1 (1.96%)	2 (3.92%)	21 (41.18%)
poi	22	10 (45.45%)	2 (9.09%)	0 (0.00%)	10 (45.45%)
abbastanza	11	4 (36.36%)	1 (9.09%)	0 (0.00%)	6 (54.55%)
anche	89	33 (37.08%)	5 (5.62%)	1 (1.12%)	53 (59.55%)
ora	17	6 (35.29%)	1 (5.88%)	1 (5.88%)	10 (58.82%)
proprio	11	5 (45.45%)	0 (0.00%)	0 (0.00%)	6 (54.55%)
quindi	35	9 (25.71%)	1 (2.86%)	0 (0.00%)	25 (71.43%)
come	16	0 (0.00%)	1 (6.25%)	1 (6.25%)	14 (87.50%)
dove	28	0 (0.00%)	1 (3.57%)	0 (0.00%)	27 (96.43%)
generalmente	13	0 (0.00%)	0 (0.00%)	0 (0.00%)	13 (100.00%)
per quanto riguarda	14	0 (0.00%)	0 (0.00%)	1 (7.14%)	13 (92.86%)
piuttosto	13	0 (0.00%)	0 (0.00%)	0 (0.00%)	13 (100.00%)
più	57	0 (0.00%)	3 (5.26%)	2 (3.51%)	52 (91.23%)
poco	63	2 (3.17%)	3 (4.76%)	0 (0.00%)	58 (92.06%)
sempre	13	1 (7.69%)	0 (0.00%)	0 (0.00%)	12 (92.31%)
soprattutto	16	1 (6.25%)	1 (6.25%)	0 (0.00%)	14 (87.50%)
a	111	0 (0.00%)	18 (16.22%)	30 (27.03%)	66 (59.46%)
con	91	0 (0.00%)	20 (21.98%)	11 (12.09%)	62 (68.13%)
da	97	0 (0.00%)	26 (26.80%)	18 (18.56%)	62 (63.92%)
di	510	2 (0.39%)	92 (18.04%)	140 (27.45%)	312 (61.18%)
e	206	17 (8.25%)	34 (16.50%)	25 (12.14%)	140 (67.96%)
in	168	6 (3.57%)	37 (22.02%)	16 (9.52%)	113 (67.26%)
per	120	0 (0.00%)	7 (5.83%)	35 (29.17%)	82 (68.33%)
su	327	4 (1.22%)	121 (37.00%)	38 (11.62%)	190 (58.10%)
verso	18	0 (0.00%)	6 (33.33%)	1 (5.56%)	12 (66.67%)

pair of sets (in this case we have only one pair), and consider the full path formed by the two paths connected by the connective node (for instance, “sulla, annuolamento, possibilità, di, scroscio, breve”).

- For each of these paths, take the signs aligned to words on different sides of the target connective, and find the shortest path between those signs in the LIS syntax tree; we call this the aligned path. For example, from “possibilità” and “scroscio” we find “POTERE, ACQUAZZONE”. If this process generates multiple paths, only the maximal ones are kept.

By looking at words within a certain distance of the connective, at their aligned signs, and at the distance between those signs in the aligned path, the program then produces one or more “translation candidates” for each occurrence of a connective:

- Sign: if the connective word is aligned to a connective sign in LIS, that is its direct translation;

- Location: if morphology variations (currently limited to the “location” attribute, see Figure 2) are present on a sign aligned to an It. word belonging to one of the examined paths, and the word is less than 2 steps away from the connective, that morphological variation in LIS may capture the function of the connective;

- Close: if two It. words are connected by a connective, and they map to signs which have a very short path between them (up to 3 nodes, including the two signs), the connective may be reflected simply in this close connection between the translated subtrees in the LIS syntax tree;

- Missing: if none of the above hypotheses are possible, we hypothesize that the connective has been simply dropped.

Table 1 shows the results of this analysis. It includes only connectives with more than 10 occurrences. For each connective and translation hypothesis, the shading of the cell is proportional to

the fraction of occurrences where that hypothesis is possible; this fraction is also given as a percent. Note that Sign, Location and Close candidates are not mutually exclusive: for instance, an occurrence of a connective might be directly aligned with a sign, but at the same time it might fit the criteria for a Location candidate. For this reason, the sum of the percents in the four columns is not necessarily 100.

k-means clustering (MacQueen, 1967; Lloyd, 1982) has been applied to the connectives, with the aforementioned fractions as the features. The resulting five clusters are represented by the row groupings in the table.

The first cluster contains words which clearly have a corresponding sign in LIS, such as “domani” (tomorrow). “Domani” and “dopodomani” are not actually connectives, while “mentre”, “o” and “però” are. It is interesting to note that, while a logician might expect “e” (and) and “o” (or) to be treated similarly, they actually work quite differently in LIS: there is a specific sign for “o”, but there is no sign for “e”. Instead, signs are simply juxtaposed in LIS where “e” would be used in Italian.

The words in the second cluster also have a direct sign translation, but they are missing in the LIS translation around half of the time. Several words represent connections with previous statements or situations, such as “ancora” (again), “invece” (instead), “ma” (but). These appear to be often dropped in LIS when they reference a previous sentence, e.g. a sentence-initial “ma”; or when they are redundant in Italian, e.g. “ma” in “ma anche” (“but also”). Therefore, we think can see two phenomena at play here: a stronger principle of economy in LIS, and a reduced number of explicit connections across sentences.

The third cluster is similar to the second cluster, but with a higher percent of dropped connectives. This is probably related to the semantics of these five words. “Abbastanza” means “quite, enough”, and in general indicates a medium quantity, not particularly large nor particularly small. It is no surprise that this word is more likely to succumb to principles of economy in language. “Anche” means “also”, and is either translated as “PURE” (also) or dropped. This does not seem to depend on the specific circumstances of its usage; rather, it seems to be largely a stylistic choice by the translator. “Proprio” (“precisely”, “just”) has a corre-

sponding sign “PROPRIO”, but since it does not convey essential information it is a good candidate for dropping. “Quindi”, meaning “therefore”, has its own sign “QUINDI”, but once again the causal relationship it conveys is usually not essential to understanding what the weather will be, and thus it is frequently dropped.

The fourth cluster consists of connectives which are largely simply dropped. Some of these are elements that just contribute to the discourse flow in Italian, such as “per quanto riguarda” (“concerning”); in fact, this connective mainly occurs in sentence-initial position in the Italian sentences in our corpus and denotes a change of topic from the previous sentence, corroborating our hypothesis of a reduced use of explicit inter-sentence connections in LIS. It may seem strange for comparative and intensity markers such as “più” (more) or “poco” (a little) to be so consistently dropped, but it turns out that intensity variations for weather phenomena are often embedded into a specific sign, for example “NUVOLOSITÀ AUMENTARE” (increasing cloud cover).

The fifth cluster contains all Italian prepositions (with 10 or more occurrences in the corpus), none of which is translated as a sign (the 6 occurrences for “in”, the 4 for “su” and the 2 for “di” are due to alignment errors). We can conclude that prepositions do not exist in LIS as parts of speech; however, the prepositions in this cluster are often associated with morphological variations in the spatial positioning of related signs, which suggests that the role associated with these prepositions in Italian is performed by these variations in LIS. The conjunction “e” (and) also ends up in this cluster, although it has 8 legitimate sign alignments with “pure” (“too”); the rest are alignment errors. Unsurprisingly, all connectives in this class also have high ratings for the “close” hypothesis.

#### 4 Rule extraction

We trained a classifier to help a LIS generator determine how an Italian connective should be translated. Because the translation pipeline we plan to integrate with is rule-based, we chose a Decision Tree as our classifier: this allows rules to be easily extracted from the classification model.

In order to identify a single class for each example, we ranked the four possible translation candidates as follows: Sign is the strongest, then Location, then Close, and finally Missing is the

$\text{child1\_align} = \text{None} \cap \text{word} = \text{Per\_quanto\_riguarda} \cap \text{parent\_align} = \text{None} \Rightarrow \text{Missing}$   
 $\text{child1\_align} = \text{None} \cap \text{word} = \text{Per\_quanto\_riguarda} \cap \text{parent\_align} = \text{PREVEDERE} \Rightarrow \text{Close}$   
 $\text{child1\_align} = \text{None} \cap \text{word} = \text{o} \Rightarrow \text{Align(O)}$   
 $\text{child1\_align} = \text{None} \cap \text{word} = \text{su} \cap \text{child2\_align\_mykind} = \text{location} \cap \text{child2\_align} = \text{SICILIA} \Rightarrow \text{Location}$

Figure 5: Some rules extracted from the decision tree

weakest. Then, each example is labeled with the strongest translation candidate available for it: thus, for example, if the connective word appears to be translated with a connective sign, and the words it connects are also aligned to signs which are close to each other syntactically, then the class is Sign, not Close.

Our training data suffers from large imbalance between the “missing” class and the others. A classifier that simply labels all examples as “missing” would have an accuracy above 60%, and in fact, that is the classifier that we obtain if we attempt to automatically optimize the parameters of a Decision Tree (DT). We also note that, for connectives where both options are possible, choosing to translate them can make the sentence more verbose, but choosing to drop them risks losing part of the sentence’s meaning: the worse risk is the latter. Following accepted practice with unbalanced datasets (Chawla et al., 2004), we rebalanced the classes by duplicating all examples of the Align, Location and Close classes, but not those of the Missing class.

On our data set of connectives with at least 10 occurrences, we trained a DT using AdaBoost (Freund and Schapire, 1997). The features include the word neighboring the connective in the Italian syntax tree, their aligned signs if any, part of speech tags, and semantic categories such as time or location. The resulting tree is very large, but we provide a few examples of the rules that can be extracted from it in Figure 5.

Bootstrap evaluation shows our DT to have an accuracy of  $83.58\% \pm 1.03\%$ . In contrast, a baseline approach of taking the most common class for each connective results in an accuracy of  $68.70\% \pm 0.88\%$ . Furthermore, the baseline classifier has abysmal recall for the Close and Location classes (0.00% and 0.85%, respectively), which our DT greatly improves upon ( $86.73\%$  and  $75.32\%$ ).

In order to estimate the impact of the lack of a LIS syntax tree in most of the corpus, we also learned and evaluated a DT using only the 1/4 of the corpus for which LIS syntax trees are available. The accuracy is  $81.44\% \pm 2.03\%$ , versus a

baseline of  $71.55\% \pm 1.74\%$ . The recall for Close and Location is  $89.22\%$  and  $73.58\%$ , vs.  $0.00\%$  and  $3.51\%$  for the baseline. These results are comparable with the those obtained on the whole corpus, confirming that linear trees are a reasonable fallback.

Both clustering and classification were performed using RapidMiner 5.3.<sup>2</sup>

## 5 Conclusions and Future Work

The small size of our corpus, with around 375 bilingual sentences, posed a large challenge to the use of statistical methods; on the other hand, having no access to a LIS speaker prevented us from simply relying on a rule-based approach. By combining syntax tree processing with several machine learning techniques, we were able to analyze the corpus and detect patterns that show linguistic substance. We have produced initial results in terms of rule extraction, and we will be integrating these rules into the full Italian-LIS translation system to produce improved translation of connectives.

## 6 Acknowledgements

This work was supported by the ATLAS project, funded by Regione Piemonte within the “CIPE 2007” framework. Partial support to the authors was also provided by awards IIS 0905593 (from the NSF) and NPRP 5-939-1-155 (from the QNRF). A special thanks to A. Mazzei (ATLAS) for his willingness to answer our email bursts. Thanks to other members of ATLAS, in particular P. Prinetto, N. Bertoldi, C. Geraci, L. Lesmo; and to C. Soria, who extracted the list of potential connectives from the Italian Treebank.

## References

Nadeem Ahmad, Davide Barberis, Nicola Garazzino, Paolo Prinetto, Umar Shoaib, and Gabriele Tiotto. 2012. A virtual character based italian sign language dictionary. In *Proceedings of the Conference Universal Learning Design*. Masaryk University.

<sup>2</sup><http://rapid-i.com/>

- Abdulaziz Almohimeed, Mike Wald, and R.I. Damper. 2011. Arabic Text to Arabic Sign Language Translation System for the Deaf and Hearing-Impaired Community. In *Proceedings of the Second Workshop on Speech and Language Processing for Assistive Technologies*, pages 101–109, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.
- Giuseppe Attardi, Felice Dell’Orletta, Maria Simi, Atanas Chanev, and Massimiliano Ciaramita. 2007. Multilingual dependency parsing and domain adaptation using DeSR. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 1112–1118.
- N Bertoldi, G Tiotto, P Prinetto, E Piccolo, F Nunnari, V Lombardo, A Mazzei, R Damiano, L Lesmo, and A Del Principe. 2010. On the creation and the annotation of a large-scale Italian-LIS parallel corpus. In *4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies, CSLT*.
- Annelies Braffort, Laurence Bolot, E Chtelat-Pel, Annick Choisier, Maxime Delorme, Michael Filhol, Jérémie Segouat, Cyril Verrecchia, Flora Badin, and Nad’ège Devos. 2010. Sign language corpora for analysis, processing and evaluation. In *Proc. of the Seventh International Conference on Language Resources and Evaluation (LREC’10)*.
- Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. 2004. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1–6.
- Massimiliano Ciaramita and Giuseppe Attardi. 2011. Dependency parsing with second-order feature maps and annotated semantic information. In H. Bunt, P. Merlo, and J. Nivre, editors, *Trends in Parsing Technology*, volume 43 of *Text, Speech and Language Technology*, pages 87–104. Springer.
- John Denero. 2007. Tailoring word alignments to syntactic machine translation. In *In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*, pages 17–24.
- Philippe Dreuw, Jens Forster, Yannick Gweth, Daniel Stein, Hermann Ney, Gregorio Martinez, Jaume Verges Llahi, Onno Crasborn, Ellen Ormel, Wei Du, et al. 2010. SignSpeak—understanding, recognition, and translation of sign languages. In *The 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies (CSLT 2010)*, pages 22–23, Valletta, Malta.
- Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 2*, pages 205–208. Association for Computational Linguistics.
- Angela Ferrari. 2008. Congiunzioni frasali, congiunzioni testuali e preposizioni: stessa logica, diversa testualità. In Emanuela Cresti, editor, *Prospettive nello studio del lessico italiano, Atti del IX Congresso della Società Internazionale di Linguistica e Filologia Italiana*, pages 411–416, Florence, Italy. Firenze University Press.
- Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.
- Carlo Geraci, Marta Gozzi, Costanza Papagno, and Carlo Cecchetto. 2008. How grammar can cope with limited short-term memory: Simultaneity and seriality in sign languages. *Cognition*, 106(2):780–804.
- Daniel Gildea. 2003. Loosely tree-based alignment for machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 80–87. Association for Computational Linguistics.
- Stuart Lloyd. 1982. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137.
- Vincenzo Lombardo, Fabrizio Nunnari, and Rossana Damiano. 2010. A virtual interpreter for the Italian Sign Language. In *Proceedings of the 10th International Conference on Intelligent Virtual Agents, IVA’10*, pages 201–207, Berlin, Heidelberg. Springer-Verlag.
- Vincenzo Lombardo, Cristina Battaglini, Rossana Damiano, and Fabrizio Nunnari. 2011. An avatar-based interface for the Italian Sign Language. In *2011 International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, pages 589–594. IEEE.
- Verónica López-Ludeña, Rubén San-Segundo, Syaheerah Lufti, Juan Manuel Lucas-Cuesta, Julián David Echevarry, and Beatriz Martínez-González. 2011. Source Language Categorization for improving a Speech into Sign Language Translation System. In *Proceedings of the Second Workshop on Speech and Language Processing for Assistive Technologies*, pages 84–93, Edinburgh, Scotland, UK, July.
- Verónica López-Ludeña, Rubén San-Segundo, Juan Manuel Montero, Ricardo Córdoba, Javier Ferreiros, and José Manuel Pardo. 2011. Automatic categorization for improving Spanish into Spanish Sign Language machine translation. *Computer Speech & Language*.
- Pengfei Lu and Matt Huenerfauth. 2010. Collecting a Motion-Capture Corpus of American Sign Language for Data-Driven Generation Research. In *Proceedings of the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive*

- Technologies*, pages 89–97, Los Angeles, California, June. Association for Computational Linguistics.
- Pengfei Lu and Matt Huenerfauth. 2012. Learning a Vector-Based Model of American Sign Language Inflecting Verbs from Motion-Capture Data. In *Proceedings of the Third Workshop on Speech and Language Processing for Assistive Technologies*, pages 66–74, Montréal, Canada, June. Association for Computational Linguistics.
- James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA.
- Jonathan May and Kevin Knight. 2007. Syntactic realignment models for machine translation. In *Proceedings of EMNLP*, pages 360–368.
- Alessandro Mazzei. 2012. Sign language generation with expert systems and ccg. In *Proceedings of the Seventh International Natural Language Generation Conference, INLG '12*, pages 105–109, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Simonetta Montemagni, Francesco Barsotti, Marco Battista, Nicoletta Calzolari, Ornella Corazzari, Alessandro Lenci, Antonio Zampolli, Francesca Fanciulli, Maria Massetani, Remo Raffaelli, et al. 2003. Building the Italian syntactic-semantic Treebank. *Treebanks*, pages 189–210.
- S. Morrissey, H. Somers, R. Smith, S. Gilchrist, and S. Dandapat. 2010. Building Sign Language Corpora for Use in Machine Translation. In *4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies (CSLT 2010)*, pages 172–178, Valletta, Malta.
- P. Prinetto, U. Shoaib, and G. Tiotto. 2011. The Italian Sign Language Sign Bank: Using WordNet for Sign Language corpus creation. In *2011 International Conference on Communications and Information Technology (ICCIT)*, pages 134–137.
- Orazio Romeo. 1991. *Dizionario dei segni: la lingua dei segni in 1400 immagini*. Zanichelli.
- Virginia Volterra. 1987. *La lingua italiana dei segni: la comunicazione visivo-gestuale dei sordi*. Il Mulino.

# Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing

David Mareček and Milan Straka

Charles University in Prague, Faculty of Mathematics and Physics  
Institute of Formal and Applied Linguistics  
Malostranské náměstí 25, 11800 Prague, Czech Republic  
{marecek, straka}@ufal.mff.cuni.cz

## Abstract

Even though the quality of unsupervised dependency parsers grows, they often fail in recognition of very basic dependencies. In this paper, we exploit a prior knowledge of STOP-probabilities (whether a given word has any children in a given direction), which is obtained from a large raw corpus using the reducibility principle. By incorporating this knowledge into Dependency Model with Valence, we managed to considerably outperform the state-of-the-art results in terms of average attachment score over 20 treebanks from CoNLL 2006 and 2007 shared tasks.

## 1 Introduction

The task of unsupervised dependency parsing (which strongly relates to the grammar induction task) has become popular in the last decade, and its quality has been greatly increasing during this period.

The first implementation of Dependency Model with Valence (DMV) (Klein and Manning, 2004) with a simple inside-outside inference algorithm (Baker, 1979) achieved 36% attachment score on English and was the first system outperforming the adjacent-word baseline.<sup>1</sup>

Current attachment scores of state-of-the-art unsupervised parsers are higher than 50% for many languages (Spitkovsky et al., 2012; Blunsom and Cohn, 2010). This is still far below the supervised approaches, but their indisputable advantage is the fact that no annotated treebanks are needed and the induced structures are not burdened by any linguistic conventions. Moreover,

<sup>1</sup>The adjacent-word baseline is a dependency tree in which each word is attached to the previous (or the following) word. The attachment score of 35.9% on all the WSJ test sentences was taken from (Blunsom and Cohn, 2010).

supervised parsers always only simulate the treebanks they were trained on, whereas unsupervised parsers have an ability to be fitted to different particular applications.

Some of the current approaches are based on the DMV, a generative model where the grammar is expressed by two probability distributions:  $P_{choose}(c_d|c_h, dir)$ , which generates a new child  $c_d$  attached to the head  $c_h$  in the direction  $dir$  (left or right), and  $P_{stop}(STOP|c_h, dir, \dots)$ , which makes a decision whether to generate another child of  $c_h$  in the direction  $dir$  or not.<sup>2</sup> Such a grammar is then inferred using sampling or variational methods.

Unfortunately, there are still cases where the inferred grammar is very different from the grammar we would expect, e.g. verbs become leaves instead of governing the sentences. Rasooli and Faili (2012) and Bisk and Hockenmaier (2012) made some efforts to boost the verbcentricity of the inferred structures; however, both of the approaches require manual identification of the POS tags marking the verbs, which renders them useless when unsupervised POS tags are employed.

The main contribution of this paper is a considerable improvement of unsupervised parsing quality by estimating the  $P_{stop}$  probabilities externally using a very large corpus, and employing this prior knowledge in the standard inference of DMV. The estimation is done using the reducibility principle introduced in (Mareček and Žabokrtský, 2012). The reducibility principle postulates that if a word (or a sequence of words) can be removed from a sentence without violating its grammatical correctness, it is a leaf (or a subtree) in its dependency structure. For the purposes of this paper, we assume the following hypothesis:

*If a sequence of words can be removed from*

<sup>2</sup>The  $P_{stop}$  probability may be conditioned by additional parameters, such as adjacency  $adj$  or fringe word  $c_f$ , which will be described in Section 4.

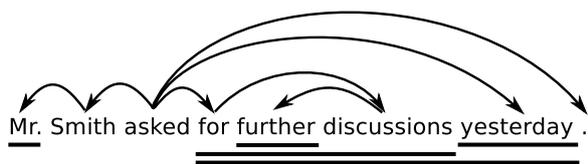


Figure 1: Example of a dependency tree. Sequences of words that can be reduced are underlined.

*a sentence without violating its grammatical correctness, no word outside the sequence depends on any word in the sequence.*

Our hypothesis is a generalization of the original hypothesis since it allows a reducible sequence to form several adjacent subtrees.

Let’s outline the connection between the  $P_{stop}$  probabilities and the property of reducibility. Figure 1 shows an example of a dependency tree. Sequences of reducible words are marked by thick lines below the sentence. Consider for example the word “*further*”. It can be removed and thus, according to our hypothesis, no other word depends on it. Therefore, we can deduce that the  $P_{stop}$  probability for such word is high both for the left and for the right direction. The phrase “*for further discussions*” is reducible as well and we can deduce that the  $P_{stop}$  of its first word (“*for*”) in the left direction is high since it cannot have any left children. We do not know anything about its right children, because they can be located within the sequence (and there is really one in Figure 1). Similarly, the word “*discussions*”, which is the last word in this sequence, cannot have any right children and we can estimate that its right  $P_{stop}$  probability is high. On the other hand, non-reducible words such, as the verb “*asked*” in our example, can have children, and therefore their  $P_{stop}$  can be estimated as low for both directions.

The most difficult task in this approach is to automatically recognize reducible sequences. This problem, together with the estimation of the stop-probabilities, is described in Section 3. Our model, not much different from the classic DMV, is introduced in Section 4. Section 5 describes the inference algorithm based on Gibbs sampling. Experiments and results are discussed in Section 6. Section 7 concludes the paper.

## 2 Related Work

**Reducibility:** The notion of reducibility belongs to the traditional linguistic criteria for recogniz-

ing dependency relations. As mentioned e.g. by Kübler et al. (2009), the head  $h$  of a construction  $c$  determines the syntactic category of  $c$  and can often replace  $c$ . In other words, the descendants of  $h$  can be often removed without making the sentence incorrect. Similarly, in the Dependency Analysis by Reduction (Lopatková et al., 2005), the authors assume that stepwise deletions of dependent elements within a sentence preserve its syntactic correctness. A similar idea of dependency analysis by splitting a sentence into all possible acceptable fragments is used by Gerdes and Kahane (2011).

We have directly utilized the aforementioned criteria for dependency relations in unsupervised dependency parsing in our previous paper (Mareček and Žabokrtský, 2012). Our dependency model contained a submodel which directly prioritized subtrees that form reducible sequences of POS tags. Reducibility scores of given POS tag sequences were estimated using a large corpus of Wikipedia articles. The weakness of this approach was the fact that longer sequences of POS tags are very sparse and no reducibility scores could be estimated for them. In this paper, we avoid this shortcoming by estimating the STOP probabilities for individual POS tags only.

Another task related to reducibility is *sentence compression* (Knight and Marcu, 2002; Cohn and Lapata, 2008), which was used for text summarization. The task is to shorten the sentences while retaining the most important pieces of information, using the knowledge of the grammar. Conversely, our task is to induce the grammar using the sentences and their shortened versions.

**Dependency Model with Valence (DMV)** has been the most popular approach to unsupervised dependency parsing in the recent years. It was introduced by Klein and Manning (2004) and further improved by Smith (2007) and Cohen et al. (2008). Headden III et al. (2009) introduce the Extended Valence Grammar and add lexicalization and smoothing. Blunsom and Cohn (2010) use tree substitution grammars, which allow learning of larger dependency fragments by employing the Pitman-Yor process. Spitkovsky et al. (2010) improve the inference using iterated learning of increasingly longer sentences. Further improvements were achieved by better dealing with punctuation (Spitkovsky et al., 2011b) and new “boundary” models (Spitkovsky et al., 2012).

**Other approaches** to unsupervised dependency parsing were described e.g. in (Søgaard, 2011), (Cohen et al., 2011), and (Bisk and Hockenmaier, 2012). There also exist “less unsupervised” approaches that utilize an external knowledge of the POS tagset. For example, Rasooli and Faili (2012) identify the last verb in the sentence, minimize its probability of reduction and thus push it to the root position. Naseem et al. (2010) make use of manually-specified universal dependency rules such as *Verb*→*Noun*, *Noun*→*Adjective*. McDonald et al. (2011) identify the POS tags by a cross-lingual transfer. Such approaches achieve better results; however, they are useless for grammar induction from plain text.

### 3 STOP-probability estimation

#### 3.1 Recognition of reducible sequences

We introduced a simple procedure for recognition of reducible sequences in (Mareček and Žabokrtský, 2012): The particular sequence of words is removed from the sentence and if the remainder of the sentence exists elsewhere in the corpus, the sequence is considered reducible. We provide an example in Figure 2. The bigram “*this weekend*” in the sentence “*The next competition is this weekend at Lillehammer in Norway.*” is reducible since the same sentence without this bigram, i.e., “*The next competition is at Lillehammer in Norway.*”, is in the corpus as well. Similarly, the prepositional phrase “*of Switzerland*” is also reducible.

It is apparent that only very few reducible sequences can be found by this procedure. If we use a corpus containing about 10,000 sentences, it is possible that we found no reducible sequences at all. However, we managed to find a sufficient amount of reducible sequences in corpora containing millions of sentences, see Section 6.1 and Table 1.

#### 3.2 Computing the STOP-probability estimations

Recall our hypothesis from Section 1: If a sequence of words is reducible, no word outside the sequence can depend on any word in the sequence. Or, in terms of dependency structure: A reducible sequence consists of one or more adjacent subtrees. This means that the first word of a reducible sequence does not have any left children and, similarly, the last word in a reducible sequence does

Martin Fourcade was sixth , maintaining his lead at the top of the overall World Cup standings , although Svendsen is now only 59 points away from the Frenchman in second . The next competition is this weekend at Lillehammer in Norway .

Larinto saw off allcomers at Kuopio with jumps of 129.5 and 124m for a total 240.9 points , just 0.1 points ahead of compatriot Matti Hautamaeki , who landed efforts of 127 and 129.5m . Third place went to Simon Ammann . Andreas Kofler , who won at the weekend at Kuusamo , was fourth but stays top of the season standings with 150 points .

Third place went to Simon Ammann of Switzerland . Ammann is currently just fifth , overall with 120 points . The next competition is at Lillehammer in Norway .

Figure 2: Example of reducible sequences of words found in a large corpus.

not have any right children. We make use of this property directly for estimating  $P_{stop}$  probabilities.

Hereinafter,  $P_{stop}^{est}(c_h, dir)$  denotes the STOP-probability we want to estimate from a large corpus;  $c_h$  is the head’s POS tag and  $dir$  is the direction in which the STOP probability is estimated. If  $c_h$  is very often in the first position of reducible sequences,  $P_{stop}^{est}(c_h, left)$  will be high. Similarly, if  $c_h$  is often in the last position of reducible sequences,  $P_{stop}^{est}(c_h, right)$  will be high.

For each POS tag  $c_h$  in the given corpus, we first compute its left and right “raw” score  $S_{stop}(c_h, left)$  and  $S_{stop}(c_h, right)$  as the relative number of times a word with POS tag  $c_h$  was in the first (or last) position in a reducible sequence found in the corpus. We do not deal with sequences longer than a trigram since they are highly biased.

$$S_{stop}(c_h, left) = \frac{\# \text{ red.seq. } [c_h, \dots] + \lambda}{\# c_h \text{ in the corpus}}$$

$$S_{stop}(c_h, right) = \frac{\# \text{ red.seq. } [\dots, c_h] + \lambda}{\# c_h \text{ in the corpus}}$$

Note that the  $S_{stop}$  scores are not probabilities. Their main purpose is to sort the POS tags according to their “reducibility”.

It may happen that for many POS tags there are no reducible sequences found. To avoid zero scores, we use a simple smoothing by adding  $\lambda$  to each count:

$$\lambda = \frac{\# \text{ all reducible sequences}}{W},$$

where  $W$  denotes the number of words in the given corpus. Such smoothing ensures that more frequent irreducible POS tags get a lower  $S_{stop}$  score than the less frequent ones.

Since reducible sequences found are very sparse, the values of  $S_{stop}(c_h, dir)$  scores are very small. To convert them to estimated probabilities  $P_{stop}^{est}(c_h, dir)$ , we need a smoothing that fulfills the following properties:

- (1)  $P_{stop}^{est}$  is a probability and therefore its value must be between 0 and 1.
- (2) The number of *no-stop* decisions (no matter in which direction) equals to  $W$  (number of words) since such decision is made before each word is generated. The number of *stop* decisions is  $2W$  since they come after generating the last children in both the directions. Therefore, the average  $P_{stop}^{est}(h, dir)$  over all words in the treebank should be  $2/3$ .

After some experimenting, we chose the following normalization formula

$$P_{stop}^{est}(c_h, dir) = \frac{S_{stop}(c_h, dir)}{S_{stop}(c_h, dir) + \nu}$$

with a normalization constant  $\nu$ . The condition (1) is fulfilled for any positive value of  $\nu$ . Its exact value is set in accordance with the requirement (2) so that the average value of  $P_{stop}^{est}$  is  $2/3$ .

$$\sum_{dir \in \{l, r\}} \sum_{c \in C} count(c) P_{stop}^{est}(c, dir) = \frac{2}{3} \cdot 2W,$$

where  $count(c)$  is the number of words with POS tag  $c$  in the corpus. We find the unique value of  $\nu$  that fulfills the previous equation numerically using a binary search algorithm.

## 4 Model

We use the standard generative Dependency Model with Valence (Klein and Manning, 2004). The generative story is the following: First, the head of the sentence is generated. Then, for each head, all its left children are generated, then the left STOP, then all its right children, and then the right STOP. When a child is generated, the algorithm immediately recurses to generate its subtree. When deciding whether to generate another child in the direction  $dir$  or the STOP symbol, we use the  $P_{stop}^{dmv}(STOP|c_h, dir, adj, c_f)$  model.

The new child  $c_d$  in the direction  $dir$  is generated according to the  $P_{choose}(c_d|c_h, dir)$  model. The probability of the whole dependency tree  $T$  is the following:

$$P_{tree}(T) = P_{choose}(head(T)|ROOT, right) \cdot P_{tree}(D(head(T)))$$

$$P_{tree}(D(c_h)) = \prod_{dir \in \{l, r\}} \prod_{\substack{c_d \in \\ deps(dir, h)}} P_{stop}^{dmv}(\neg STOP|c_h, dir, adj, c_f) P_{choose}(c_d|c_h, dir) P_{tree}(D(c_d)) P_{stop}^{dmv}(STOP|c_h, dir, adj, c_f),$$

where  $P_{tree}(D(c_h))$  is probability of the subtree governed by  $h$  in the tree  $T$ .

The set of features on which the  $P_{stop}^{dmv}$  and  $P_{choose}$  probabilities are conditioned varies among the previous works. Our  $P_{stop}^{dmv}$  depends on the head POS tag  $c_h$ , direction  $dir$ , adjacency  $adj$ , and fringe POS tag  $c_f$  (described below). The use of adjacency is standard in DMV and enables us to have different  $P_{stop}^{dmv}$  for situations when no child was generated so far ( $adj = 1$ ). That is,  $P_{stop}^{dmv}(c_h, dir, adj = 1, c_f)$  decides whether the word  $c_h$  has any children in the direction  $dir$  at all, whereas  $P_{stop}^{dmv}(h, dir, adj = 0, c_f)$  decides whether another child will be generated next to the already generated one. This distinction is of crucial importance for us: although we know how to estimate the STOP probabilities for  $adj = 1$  from large data, we do not know anything about the STOP probabilities for  $adj = 0$ .

The last factor  $c_f$ , called fringe, is the POS tag of the previously generated sibling in the current direction  $dir$ . If there is no such sibling (in case  $adj = 1$ ), the head  $c_h$  is used as the fringe  $c_f$ . This is a relatively novel idea in DMV, introduced by Spitzkovsky et al. (2012). We decided to use the fringe word in our model since it gives slightly better results.

We assume that the distributions of  $P_{choose}$  and  $P_{stop}^{dmv}$  are good if the majority of the probability mass is concentrated on few factors; therefore, we apply a Chinese Restaurant process (CRP) on them.

The probability of generating a new child node  $c_d$  attached to  $c_h$  in the direction  $dir$  given the history (all the nodes we have generated so far) is

computed using the following formula:

$$P_{choose}(c_d|c_h, dir) = \frac{\alpha_c \frac{1}{|C|} + count^-(c_d, c_h, dir)}{\alpha_c + count^-(c_h, dir)},$$

where  $count^-(c_d, c_h, dir)$  denotes the number of times a child node  $c_d$  has been attached to  $c_h$  in the direction  $dir$  in the history. Similarly,  $count^-(c_h, dir)$  is the number of times something has been attached to  $c_h$  in the direction  $dir$ . The  $\alpha_c$  is a hyperparameter and  $|C|$  is the number of distinct POS tags in the corpus.<sup>3</sup>

The STOP probability is computed in a similar way:

$$P_{stop}^{dmv}(STOP|c_h, dir, adj, c_f) = \frac{\alpha_s \frac{2}{3} + count^-(STOP, c_h, dir, adj, c_f)}{\alpha_s + count^-(c_h, dir, adj, c_f)}$$

where  $count^-(STOP, c_h, dir, adj, c_f)$  is the number of times a head  $c_h$  had the last child  $c_f$  in the direction  $dir$  in the history.

The contribution of this paper is the inclusion of the stop-probability estimates into the DMV. Therefore, we introduce a new model  $P_{stop}^{dmv+est}$ , in which the probability based on the previously generated data is linearly combined with the probability estimates based on large corpora (Section 3).

$$P_{stop}^{dmv+est}(STOP|c_h, dir, 1, c_f) = (1 - \beta) \cdot \frac{\alpha_s \frac{2}{3} + count^-(STOP, c_h, dir, 1, c_f)}{\alpha_s + count^-(c_h, dir, 1, c_f)} + \beta \cdot P_{stop}^{est}(c_h, dir)$$

$$P_{stop}^{dmv+est}(STOP|c_h, dir, 0, c_f) = P_{stop}^{dmv}(STOP|c_h, dir, 0, c_f)$$

The hyperparameter  $\beta$  defines the ratio between the CRP-based and estimation-based probability. The definition of the  $P_{stop}^{dmv+est}$  for  $adj = 0$  equals the basic  $P_{stop}^{dmv}$  since we are able to estimate only the probability whether a particular head POS tag  $c_h$  can or cannot have children in a particular direction, i.e if  $adj = 1$ .

<sup>3</sup>The number of classes  $|C|$  is often used in the denominator. We decided to put its reverse value into the numerator since we observed such model to perform better for a constant value of  $\alpha_c$  over different languages and tagsets.

Finally, we obtain the probability of the whole generated treebank as a product over the trees:

$$P_{treebank} = \prod_{T \in treebank} P_{tree}(T).$$

An important property of the CRP is the fact that the factors are exchangeable – i.e. no matter how the trees are ordered in the treebank, the  $P_{treebank}$  is always the same.

## 5 Inference

We employ the Gibbs sampling algorithm (Gilks et al., 1996). Unlike in (Mareček and Žabokrtský, 2012), where edges were sampled individually, we sample whole trees from all possibilities on a given sentence using dynamic programming. The algorithm works as follows:

1. A random projective dependency tree is assigned to each sentence in the corpus.
2. *Sampling*: We go through the sentences in a random order. For each sentence, we sample a new dependency tree based on all other trees that are currently in the corpus.
3. Step 2 is repeated in many iterations. In this work, the number of iterations was set to 1000.
4. After the burn-in period (which was set to the first 500 iterations), we start collecting counts of edges between particular words that appeared during the sampling.
5. *Parsing*: Based on the collected counts, we compute the final dependency trees using the Chu-Liu/Edmonds’ algorithm (1965) for finding maximum directed spanning trees.

### 5.1 Sampling

Our goal is to sample a new projective dependency tree  $T$  with probability proportional to  $P_{tree}(T)$ . Since the factors are exchangeable, we can deal with any tree as if it was the last one in the corpus.

We use dynamic programming to sample a tree with  $N$  nodes in  $\mathcal{O}(N^4)$  time. Nevertheless, we sample trees using a modified probability  $P'_{tree}(T)$ . In  $P_{tree}(T)$ , the probability of an edge depends on counts of all other edges, including the edges in the same tree. We instead use  $P'_{tree}(T)$ , where the counts are computed using only the other trees in the corpus, i.e., probabilities

of edges of  $T$  are independent. There is a standard way to sample using the real  $P_{tree}(T)$  – we can use  $P'_{tree}(T)$  as a proposal distribution in the Metropolis-Hastings algorithm (Hastings, 1970), which then produces trees with probabilities proportional to  $P_{tree}(T)$  using acceptance-rejection scheme. We do not take this approach and we sample proportionally to  $P'_{tree}(T)$  only, because we believe that for large enough corpora, the two distributions are nearly identical.

To sample a tree containing words  $w_1, \dots, w_N$  with probability proportional to  $P'_{tree}(T)$ , we first compute three tables:

- $t_i(g, i, j)$  for  $g < i$  or  $g > j$  is the sum of probabilities of any tree on words  $w_i, \dots, w_j$  whose root is a child of  $w_g$ , but not an outermost child in its direction;
- $t_o(g, i, j)$  is the same, but the tree is the outermost child of  $w_g$ ;
- $f_o(g, i, j)$  for  $g < i$  or  $g > j$  is the sum of probabilities of any forest on words  $w_i, \dots, w_j$ , such that all the trees are children of  $w_g$  and are the outermost children of  $w_g$  in their direction.

All the probabilities are computed using the  $P'_{tree}$ . If we compute the tables inductively from the smallest trees to the largest trees, we can precompute all the  $\mathcal{O}(N^3)$  values in  $\mathcal{O}(N^4)$  time.

Using these tables, we sample the tree recursively, starting from the root. At first, we sample the root  $r$  proportionally to the probability of a tree with the root  $r$ , which is a product of the probability of left children of  $r$  and right children of  $r$ . The probability of left children of  $r$  is either  $P'_{stop}(STOP|r, left)$  if  $r$  has no children, or  $P'_{stop}(\neg STOP|r, left)f_o(r, 1, r-1)$  otherwise; the probability of right children is analogous.

After sampling the root, we sample the ranges of its left children, if any. We sample the first left child range  $l_1$  proportionally either to  $t_o(r, 1, r-1)$  if  $l_1 = 1$ , or to  $t_i(r, l_1, r-1)f_o(r, 1, l_1-1)$  if  $l_1 > 1$ . Then we sample the second left child range  $l_2$  proportionally either to  $t_o(r, 1, l_1-1)$  if  $l_2 = 1$ , or to  $t_i(r, l_2, l_1-1)f_o(r, 1, l_2-1)$  if  $l_2 > 1$ , and so on, while there are any left children. The right children ranges are sampled similarly. Finally, we recursively sample the children, i.e., their roots, their children and so on. It is simple to verify using the definition of  $P_{tree}$  that the described method indeed samples trees proportionally to  $P'_{tree}$ .

## 5.2 Parsing

Beginning the 500th iteration, we start collecting counts of individual dependency edges during the remaining iterations. After each iteration is finished (all the trees in the corpus are re-sampled), we increment the counter of all directed pairs of nodes which are connected by a dependency edge in the current trees.

After the last iteration, we use these collected counts as weights and compute maximum directed spanning trees using the Chu-Liu/Edmonds' algorithm (Chu and Liu, 1965). Therefore, the resulting trees consist of edges maximizing the sum of individual counts:

$$T_{MST} = \arg \max_T \sum_{e \in T} count(e)$$

It is important to note that the MST algorithm may produce non-projective trees. Even if we average the strictly projective dependency trees, some non-projective edges may appear in the result. This might be an advantage since correct non-projective edges can be predicted; however, this relaxation may introduce mistakes as well.

## 6 Experiments

### 6.1 Data

We use two types of resources in our experiments. The first type are CoNLL treebanks from the year 2006 (Buchholz and Marsi, 2006) and 2007 (Nivre et al., 2007), which we use for inference and for evaluation. As is the standard practice in unsupervised parsing evaluation, we removed all punctuation marks from the trees. In case a punctuation node was not a leaf, its children are attached to the parent of the removed node.

For estimating the STOP probabilities (Section 3), we use the Wikipedia articles from W2C corpus (Majliš and Žabokrtský, 2012), which provide sufficient amount of data for our purposes. Statistics across languages are shown in Table 1.

The Wikipedia texts were automatically tokenized and segmented to sentences so that their tokenization was similar to the one in the CoNLL evaluation treebanks. Unfortunately, we were not able to find any segmenter for Chinese that would produce a desired segmentation; therefore, we removed Chinese from evaluation.

The next step was to provide the Wikipedia texts with POS tags. We employed the TnT tagger (Brants, 2000) which was trained on the re-

language	tokens (mil.)	red. seq.	language	tokens (mil.)	red. seq.
Arabic	19.7	546	Greek	20.9	1037
Basque	14.1	645	Hungarian	26.3	2237
Bulgarian	18.8	1808	Italian	39.7	723
Catalan	27.0	712	Japanese	2.6	31
Czech	20.3	930	Portuguese	31.7	4765
Danish	15.9	576	Slovenian	13.7	513
Dutch	27.1	880	Spanish	53.4	1156
English	85.0	7603	Swedish	19.2	481
German	56.9	1488	Turkish	16.5	5706

Table 1: Wikipedia texts statistics: total number of tokens and number of reducible sequences found in them.

spective CoNLL training data. The quality of such tagging is not very high since we do not use any lexicons or pretrained models. However, it is sufficient for obtaining usable stop probability estimates.

## 6.2 Estimated STOP probabilities

We applied the algorithm described in Section 3 on the prepared Wikipedia corpora and obtained the stop-probabilities  $P_{stop}^{est}$  in both directions for all the languages and their POS tags. To evaluate the quality of our estimations, we compare them with  $P_{stop}^{tb}$ , the stop probabilities computed directly on the evaluation treebanks. The comparisons on five selected languages are shown in Figure 3. The individual points represent the individual POS tags, their size (area) shows their frequency in the particular treebank. The y-axis shows the stop probabilities estimated on Wikipedia by our algorithm, while the x-axis shows the stop probabilities computed on the evaluation CoNLL data. Ideally, the computed and estimated stop probabilities should be the same, i.e. all the points should be on the diagonal.

Let’s focus on the graphs for English. Our method correctly recognizes that adverbs RB and adjectives JJ are often leaves (their stop probabilities in both directions are very high). Moreover, the estimates for RB are even higher than JJ, which will contribute to attaching adverbs to adjectives and not reversely. Nouns (NN, NNS) are somewhere in the middle, the stop probabilities for proper nouns (NNP) are estimated higher, which is correct since they have much less modifiers than the common nouns NN. The determiners are more problematic. Their estimated stop probability is not very high (about 0.65), while in the real treebank they are almost always leaves.

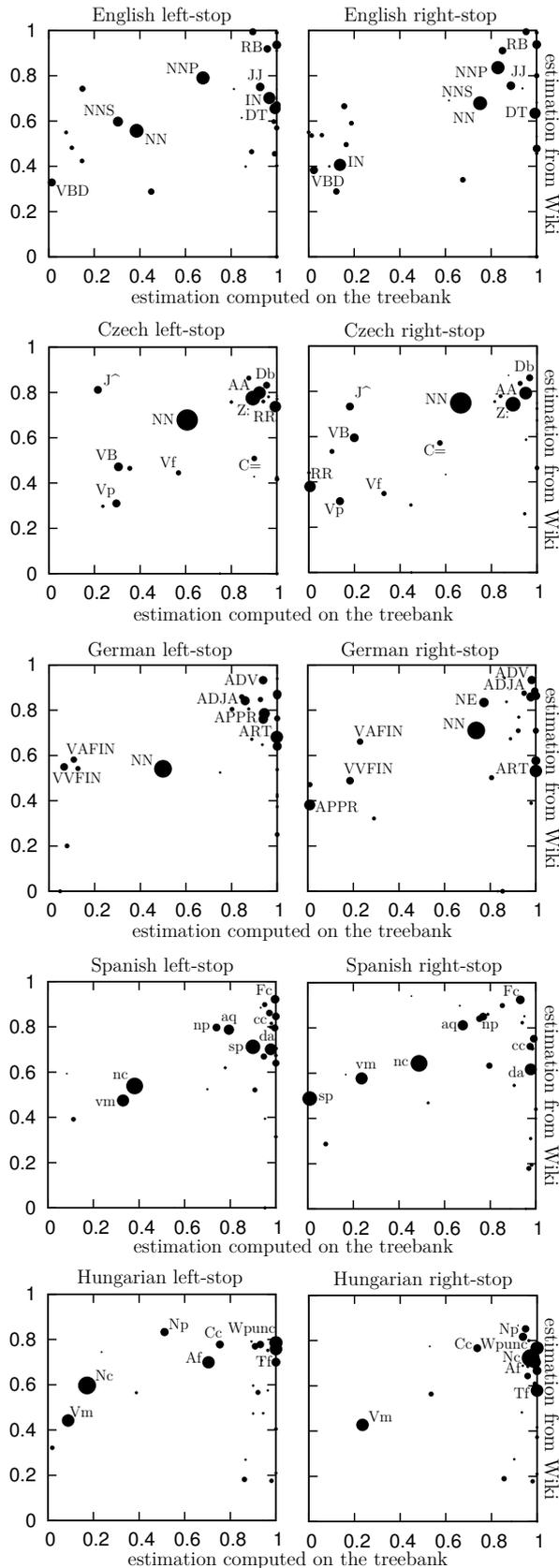


Figure 3: Comparison of  $P_{stop}^{est}$  probabilities estimated from raw Wikipedia corpora (y-axis) and of  $P_{stop}^{tb}$  probabilities computed from CoNLL treebanks (x-axis). The area of each point shows the relative frequency of an individual tag.

This is caused by the fact that determiners are often obligatory in English and cannot be simply removed as, e.g., adjectives. The stop probabilities of prepositions (IN) are also very well recognized. While their left-stop is very probable (prepositions always start prepositional phrases), their right-stop probability is very low. The verbs (the most frequent verbal tag is VBD) have very low both right and left-stop probabilities. Our estimation assigns them the stop probability about 0.3 in both directions. This is quite high, but still, it is one of the lowest among other more frequent tags, and thus verbs tend to be the roots of the dependency trees. We could make similar analyses for other languages, but due to space reasons we only provide graphs for Czech, German, Spanish, and Hungarian in Figure 3.

### 6.3 Settings

After a manual tuning, we have set our hyperparameters to the following values:

$$\alpha_c = 50, \quad \alpha_s = 1, \quad \beta = 1/3$$

We have also found that the Gibbs sampler does not always converge to a similar grammar. For a couple of languages, the individual runs end up with very different trees. To prevent such differences, we run each inference 50 times and take the run with the highest final  $P_{treebank}$  (see Section 4) for the evaluation.

### 6.4 Results

Table 2 shows the results of our unsupervised parser and compares them with results previously reported in other works. In order to see the impact of using the estimated stop probabilities (using model  $P_{stop}^{dmv+est}$ ), we provide results for classical DMV (using model  $P_{stop}^{dmv}$ ) as well. We do not provide results for Chinese since we do not have any appropriate tokenizer at our disposal (see Section 3), and also for Turkish from CoNLL 2006 since the data is not available to us.

We now focus on the third and fourth column of Table 2. The addition of estimated stop probabilities based on large corpora improves the parsing accuracy on 15 out of 20 treebanks. In many cases, the improvement is substantial, which means that the estimated stop probabilities forced the model to completely rebuild the structures. For example, in Bulgarian, if the  $P_{stop}^{dmv}$  model is used, all the prepositions are leaves and the verbs seldom govern sentences. If the  $P_{stop}^{dmv+est}$  model is used, prepositions correctly govern nouns and

verbs move to roots. We observe similar changes on Swedish as well. Unfortunately, there are also negative examples, such as Hungarian, where the addition of the estimated stop probabilities decreases the attachment score from 60.1% to 34%. This is probably caused by not very good estimates of the right-stop probability (see the last graph in Figure 3). Nevertheless, the estimated stop probabilities increase the average score over all the treebanks by more than 12% and therefore prove its usefulness.

In the last two columns of Table 2, we provide results of two other works reported in the last year. The first one (*spi12*) is the DMV-based grammar inducer by Spitzkovsky et al. (2012),<sup>4</sup> the second one (*mar12*) is our previous work (Mareček and Žabokrtský, 2012). Comparing with (Spitzkovsky et al., 2012), our parser reached better accuracy on 12 out of 20 treebanks. Although this might not seem as a big improvement, if we compare the average scores over the treebanks, our system significantly wins by more than 6%. The second system (*mar12*) outperforms our parser only on one treebank (on Italian by less than 3%) and its average score over all the treebanks is only 40%, i.e., more than 8% lower than the average score of our parser.

To see the theoretical upper bound of our model performance, we replaced the  $P_{stop}^{est}$  estimates by the  $P_{stop}^{tb}$  estimates computed from the evaluation treebanks and run the same inference algorithm with the same setting. The average attachment score of such reference DMV is almost 65%. This shows a huge space in which the estimation of STOP probabilities could be further improved.

## 7 Conclusions and Future Work

In this work, we studied the possibility of estimating the DMV stop-probabilities from a large raw corpus. We proved that such prior knowledge about stop-probabilities incorporated into the standard DMV model significantly improves the unsupervised dependency parsing and, since we are not aware of any other fully unsupervised dependency parser with higher average attachment score over CoNLL data, we state that we reached a new state-of-the-art result.<sup>5</sup>

<sup>4</sup>Possibly the current state-of-the-art results. They were compared with many previous works.

<sup>5</sup>A possible competitive work may be the work by Blunsom and Cohn (2010), who reached 55% accuracy on English as well. However, they do not provide scores measured on other CoNLL treebanks.

CoNLL		this work			other systems	
language	year	$P_{stop}^{dmv}$	$P_{stop}^{dmv+est}$	reference $P_{stop}^{dmv+tb}$	spi12	mar12
Arabic	06	10.6 ( $\pm 8.7$ )	<b>38.2 (<math>\pm 0.5</math>)</b>	61.2	10.9	26.5
Arabic	07	22.0 ( $\pm 0.1$ )	35.3 ( $\pm 0.2$ )	65.3	<b>44.9</b>	27.9
Basque	07	<b>41.1 (<math>\pm 0.2</math>)</b>	35.5 ( $\pm 0.2$ )	52.3	33.3	26.8
Bulgarian	06	25.9 ( $\pm 1.4$ )	54.9 ( $\pm 0.2$ )	73.2	<b>65.2</b>	46.0
Catalan	07	34.9 ( $\pm 3.4$ )	<b>67.0 (<math>\pm 1.7</math>)</b>	72.0	62.1	47.0
Czech	06	32.3 ( $\pm 3.8$ )	52.4 ( $\pm 5.2$ )	64.0	<b>55.1</b>	49.5
Czech	07	32.9 ( $\pm 0.8$ )	51.9 ( $\pm 5.2$ )	62.1	<b>54.2</b>	48.0
Danish	06	30.8 ( $\pm 4.3$ )	<b>41.6 (<math>\pm 1.1</math>)</b>	60.0	22.2	38.6
Dutch	06	25.7 ( $\pm 5.7$ )	<b>47.5 (<math>\pm 0.4</math>)</b>	58.9	46.6	44.2
English	07	36.5 ( $\pm 5.9$ )	<b>55.4 (<math>\pm 0.2</math>)</b>	63.7	29.6	49.2
German	06	29.9 ( $\pm 4.6$ )	<b>52.4 (<math>\pm 0.7</math>)</b>	65.5	39.1	44.8
Greek	07	<b>42.5 (<math>\pm 6.0</math>)</b>	26.3 ( $\pm 0.1$ )	64.7	26.9	20.2
Hungarian	07	<b>60.8 (<math>\pm 0.2</math>)</b>	34.0 ( $\pm 0.3$ )	68.3	58.2	51.8
Italian	07	34.5 ( $\pm 0.3$ )	39.4 ( $\pm 0.5$ )	64.5	40.7	<b>43.3</b>
Japanese	06	<b>64.8 (<math>\pm 3.4</math>)</b>	61.2 ( $\pm 1.7$ )	76.4	22.7	50.8
Portuguese	06	35.7 ( $\pm 4.3$ )	69.6 ( $\pm 0.1$ )	77.3	<b>72.4</b>	50.6
Slovenian	06	<b>50.1 (<math>\pm 0.2</math>)</b>	35.7 ( $\pm 0.2$ )	50.2	35.2	18.1
Spanish	06	38.1 ( $\pm 5.9$ )	<b>61.1 (<math>\pm 0.1</math>)</b>	65.6	28.2	51.9
Swedish	06	28.0 ( $\pm 2.3$ )	<b>54.5 (<math>\pm 0.4</math>)</b>	61.6	50.7	48.2
Turkish	07	51.6 ( $\pm 5.5$ )	<b>56.9 (<math>\pm 0.2</math>)</b>	67.0	44.8	15.7
Average:		36.4	<b>48.7</b>	64.7	42.2	40.0

Table 2: Attachment scores on CoNLL 2006 and 2007 data. Standard deviations are provided in brackets. DMV model using standard  $P_{stop}^{dmv}$  probability is compared with DMV with  $P_{stop}^{dmv+est}$ , which incorporates STOP estimations based on reducibility principle. The reference DMV uses  $P_{stop}^{tb}$ , which are computed directly on the treebanks. The results reported in previous works by Spitkovsky et al. (2012), and Mareček and Žabokrtský (2012) follows.

In future work, we would like to focus on unsupervised parsing without gold POS tags (see e.g. Spitkovsky et al. (2011a) and Christodoulopoulos et al. (2012)). We suppose that many of the current works on unsupervised dependency parsers use gold POS tags only as a simplification of this task, and that the ultimate purpose of this effort is to develop a fully unsupervised induction of linguistic structure from raw texts that would be useful across many languages, domains, and applications.

The software which implements the algorithms described in this paper, together with  $P_{stop}^{est}$  estimations computed on Wikipedia texts, can be downloaded at

<http://ufal.mff.cuni.cz/~marecek/udp/>.

## Acknowledgments

This work has been supported by the AMALACH grant (DF12P01OVV02) of the Ministry of Culture of the Czech Republic.

Data and some tools used as a prerequisite for the research described herein have been provided by the LINDAT/CLARIN Large Infrastructural project, No. LM2010013 of the Ministry of Education, Youth and Sports of the Czech Republic.

We would like to thank Martin Popel, Zdeněk Žabokrtský, Rudolf Rosa, and three anonymous reviewers for many useful comments on the manuscript of this paper.

## References

- James K. Baker. 1979. Trainable grammars for speech recognition. In *Speech communication papers presented at the 97th Meeting of the Acoustical Society*, pages 547–550.
- Yonatan Bisk and Julia Hockenmaier. 2012. Induction of linguistic structure with combinatory categorial grammars. *The NAACL-HLT Workshop on the Induction of Linguistic Structure*, page 90.
- Phil Blunsom and Trevor Cohn. 2010. Unsupervised induction of tree substitution grammars for dependency parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pages 1204–1213, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Thorsten Brants. 2000. TnT - A Statistical Part-of-Speech Tagger. *Proceedings of the sixth conference on Applied natural language processing*, page 8.
- Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X '06*, pages 149–164, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2012. Turning the pipeline into a loop: Iterated unsupervised dependency parsing and PoS induction. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 96–99, June.
- Y. J. Chu and T. H. Liu. 1965. On the Shortest Arborescence of a Directed Graph. *Science Sinica*, 14:1396–1400.

- Shay B. Cohen, Kevin Gimpel, and Noah A. Smith. 2008. Logistic normal priors for unsupervised probabilistic grammar induction. In *Neural Information Processing Systems*, pages 321–328.
- Shay B. Cohen, Dipanjan Das, and Noah A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 50–61, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 137–144, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kim Gerdes and Sylvain Kahane. 2011. Defining dependencies (and constituents). In *Proceedings of Dependency Linguistics 2011*, Barcelona.
- Walter R. Gilks, S. Richardson, and David J. Spiegelhalter. 1996. *Markov chain Monte Carlo in practice*. Interdisciplinary statistics. Chapman & Hall.
- W. Keith Hastings. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):pp. 97–109.
- William P. Headden III, Mark Johnson, and David McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 101–109, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artif. Intell.*, 139(1):91–107, July.
- Sandra Kübler, Ryan T. McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.
- Markéta Lopatková, Martin Plátek, and Vladislav Kuboň. 2005. Modeling syntax of free word-order languages: Dependency analysis by reduction. In Václav Matoušek, Pavel Mautner, and Tomáš Pavelka, editors, *Lecture Notes in Artificial Intelligence, Proceedings of the 8th International Conference, TSD 2005*, volume 3658 of *Lecture Notes in Computer Science*, pages 140–147, Berlin / Heidelberg. Springer.
- Martin Majliš and Zdeněk Žabokrtský. 2012. Language richness of the web. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey, May. European Language Resources Association (ELRA).
- David Mareček and Zdeněk Žabokrtský. 2012. Exploiting reducibility in unsupervised dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 297–307, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1234–1244, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June. Association for Computational Linguistics.
- Mohammad Sadeh Rasooli and Hesham Faili. 2012. Fast unsupervised dependency parsing with arc-standard transitions. In *Proceedings of ROBUST-UNSUP*, pages 1–9.
- Noah Ashton Smith. 2007. *Novel estimation methods for unsupervised discovery of latent structure in natural language text*. Ph.D. thesis, Baltimore, MD, USA. AAI3240799.
- Anders Søgaard. 2011. From ranked words to dependency trees: two-stage unsupervised non-projective dependency parsing. In *Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing*, TextGraphs-6, pages 60–68, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010. From baby steps to leapfrog: how "less is more" in unsupervised dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 751–759, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Valentin I. Spitkovsky, Hiyan Alshawi, Angel X. Chang, and Daniel Jurafsky. 2011a. Unsupervised dependency parsing without gold part-of-speech tags. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.
- Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011b. Punctuation: Making a point in unsupervised dependency parsing. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL-2011)*.
- Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2012. Three Dependency-and-Boundary Models for Grammar Induction. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*.

# Transfer Learning for Constituency-Based Grammars

Yuan Zhang, Regina Barzilay

Massachusetts Institute of Technology  
{yuanzh, regina}@csail.mit.edu

Amir Globerson

The Hebrew University  
gamir@cs.huji.ac.il

## Abstract

In this paper, we consider the problem of cross-formalism transfer in parsing. We are interested in parsing constituency-based grammars such as HPSG and CCG using a small amount of data specific for the target formalism, and a large quantity of coarse CFG annotations from the Penn Treebank. While all of the target formalisms share a similar basic syntactic structure with Penn Treebank CFG, they also encode additional constraints and semantic features. To handle this apparent discrepancy, we design a probabilistic model that jointly generates CFG and target formalism parses. The model includes features of both parses, allowing transfer between the formalisms, while preserving parsing efficiency. We evaluate our approach on three constituency-based grammars — CCG, HPSG, and LFG, augmented with the Penn Treebank-1. Our experiments show that across all three formalisms, the target parsers significantly benefit from the coarse annotations.<sup>1</sup>

## 1 Introduction

Over the last several decades, linguists have introduced many different grammars for describing the syntax of natural languages. Moreover, the ongoing process of developing new formalisms is intrinsic to linguistic research. However, before these grammars can be used for statistical parsing, they require annotated sentences for training. The difficulty of obtaining such annotations is a key limiting factor that inhibits the effective use of these grammars.

<sup>1</sup>The source code for the work is available at <http://groups.csail.mit.edu/rbg/code/grammar/acl2013>.

The standard solution to this bottleneck has relied on manually crafted transformation rules that map readily available syntactic annotations (e.g. the Penn Treebank) to the desired formalism. Designing these transformation rules is a major undertaking which requires multiple correction cycles and a deep understanding of the underlying grammar formalisms. In addition, designing these rules frequently requires external resources such as Wordnet, and even involves correction of the existing treebank. This effort has to be repeated for each new grammar formalism, each new annotation scheme and each new language.

In this paper, we propose an alternative approach for parsing constituency-based grammars. Instead of using manually-crafted transformation rules, this approach relies on a small amount of annotations in the target formalism. Frequently, such annotations are available in linguistic texts that introduce the formalism. For instance, a textbook on HPSG (Pollard and Sag, 1994) illustrates grammatical constructions using about 600 examples. While these examples are informative, they are not sufficient for training. To compensate for the annotation sparsity, our approach utilizes coarsely annotated data readily available in large quantities. A natural candidate for such coarse annotations is context-free grammar (CFG) from the Penn Treebank, while the target formalism can be any constituency-based grammars, such as Combinatory Categorical Grammar (CCG) (Steedman, 2001), Lexical Functional Grammar (LFG) (Bresnan, 1982) or Head-Driven Phrase Structure Grammar (HPSG) (Pollard and Sag, 1994). All of these formalisms share a similar basic syntactic structure with Penn Treebank CFG. However, the target formalisms also encode additional constraints and semantic features. For instance, Penn Treebank annotations do not make an explicit distinction between complement and adjunct, while all the above grammars mark these

roles explicitly. Moreover, even the identical syntactic information is encoded differently in these formalisms. An example of this phenomenon is the marking of subject. In LFG, this information is captured in the mapping equation, namely  $\uparrow \text{SBJ} = \downarrow$ , while Penn Treebank represents it as a functional tag, such as NP-SBJ. Figure 1 shows derivations in the three target formalisms we consider, as well as a CFG derivation. We can see that the derivations of these formalisms share the same basic structure, while the formalism-specific information is mainly encoded in the lexical entries and node labels.

To enable effective transfer the model has to identify shared structural components between the formalisms despite the apparent differences. Moreover, we do not assume parallel annotations. To this end, our model jointly parses the two corpora according to the corresponding annotations, enabling transfer via parameter sharing. In particular, we augment each target tree node with hidden variables that capture the connection to the coarse annotations. Specifically, each node in the target tree has two labels: an entry which is specific to the target formalism, and a latent label containing a value from the Penn Treebank tagset, such as NP (see Figure 2). This design enables us to represent three types of features: the target formalism-specific features, the coarse formalism features, and features that connect the two. This modeling approach makes it possible to perform transfer to a range of target formalisms, without manually drafting formalism-specific rules.

We evaluate our approach on three constituency-based grammars — CCG, HPSG, and LFG. As a source of coarse annotations, we use the Penn Treebank.<sup>2</sup> Our results clearly demonstrate that for all three formalisms, parsing accuracy can be improved by training with additional coarse annotations. For instance, the model trained on 500 HPSG sentences achieves labeled dependency F-score of 72.3%. Adding 15,000 Penn Treebank sentences during training leads to 78.5% labeled dependency F-score, an absolute improvement of 6.2%. To achieve similar performance in the absence of coarse annotations, the parser has to be trained on about 1,500 sentences, namely three times what is needed when using coarse annotations. Similar results are

<sup>2</sup>While the Penn Treebank-2 contains richer annotations, we decided to use the Penn Treebank-1 to demonstrate the feasibility of transfer from coarse annotations.

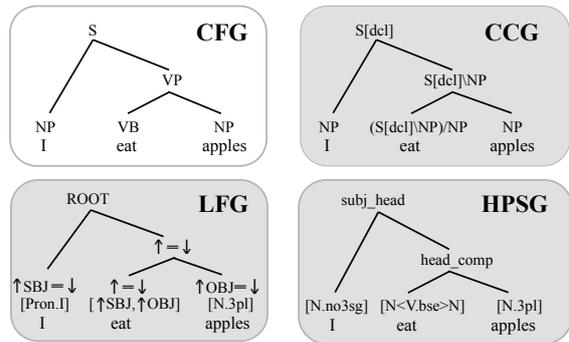


Figure 1: Derivation trees for CFG as well as CCG, HPSG and LFG formalisms.

also observed on CCG and LFG formalisms.

## 2 Related Work

Our work belongs to a broader class of research on transfer learning in parsing. This area has garnered significant attention due to the expense associated with obtaining syntactic annotations. Transfer learning in parsing has been applied in different contexts, such as multilingual learning (Snyder et al., 2009; Hwa et al., 2005; McDonald et al., 2006; McDonald et al., 2011; Jiang and Liu, 2009), domain adaptation (McClosky et al., 2010; Dredze et al., 2007; Blitzer et al., 2006), and cross-formalism transfer (Hockenmaier and Steedman, 2002; Miyao et al., 2005; Cahill et al., 2002; Riezler et al., 2002; Chen and Shanker, 2005; Candito et al., 2010).

There have been several attempts to map annotations in coarse grammars like CFG to annotations in richer grammar, like HPSG, LFG, or CCG. Traditional approaches in this area typically rely on manually specified rules that encode the relation between the two formalisms. For instance, mappings may specify how to convert traces and functional tags in Penn Treebank to the f-structure in LFG (Cahill, 2004). These conversion rules are typically utilized in two ways: (1) to create a new treebank which is consequently used to train a parser for the target formalism (Hockenmaier and Steedman, 2002; Clark and Curran, 2003; Miyao et al., 2005; Miyao and Tsujii, 2008), (2) to translate the output of a CFG parser into the target formalism (Cahill et al., 2002).

The design of these rules is a major linguistic and computational undertaking, which requires multiple iterations over the data to increase coverage (Miyao et al., 2005; Oepen et al., 2004). By nature, the mapping rules are formalism spe-

cific and therefore not transferable. Moreover, frequently designing such mappings involves modification to the original annotations. For instance, Hockenmaier and Steedman (2002) made thousands of POS and constituent modifications to the Penn Treebank to facilitate transfer to CCG. More importantly, in some transfer scenarios, deterministic rules are not sufficient, due to the high ambiguity inherent in the mapping. Therefore, our work considers an alternative set-up for cross-formalism transfer where a small amount of annotations in the target formalism is used as an alternative to using deterministic rules.

The limitation of deterministic transfer rules has been recognized in prior work (Riezler et al., 2002). Their method uses a hand-crafted LFG parser to create a set of multiple parsing candidates for a given sentence. Using the partial mapping from CFG to LFG as the guidance, the resulting trees are ranked based on their consistency with the labeled LFG bracketing imported from CFG. In contrast to this method, we neither require a parser for the target formalism, nor manual rules for partial mapping. Consequently, our method can be applied to many different target grammar formalisms without significant engineering effort for each one. The utility of coarse-grained treebanks is determined by the degree of structural overlap with the target formalism.

### 3 The Learning Problem

Recall that our goal is to learn how to parse the target formalisms while using two annotated sources: a small set of sentences annotated in the target formalism (e.g., CCG), and a large set of sentences with coarse annotations. For the latter, we use the CFG parses from the Penn Treebank. For simplicity we focus on the CCG formalism in what follows. We also generalize our model to other formalisms, as explained in Section 5.4.

Our notations are as follows: an input sentence is denoted by  $S$ . A CFG parse is denoted by  $y_{CFG}$  and a CCG parse is denoted by  $y_{CCG}$ . Clearly the set of possible values for  $y_{CFG}$  and  $y_{CCG}$  is determined by  $S$  and the grammar. The training set is a set of  $N$  sentences  $S_1, \dots, S_N$  with CFG parses  $y_{CFG}^1, \dots, y_{CFG}^N$ , and  $M$  sentences  $\bar{S}_1, \dots, \bar{S}_M$  with CCG parses  $y_{CCG}^1, \dots, y_{CCG}^M$ . It is important to note that we do not assume we have parallel data for CCG and CFG.

Our goal is to use such a corpus for learning

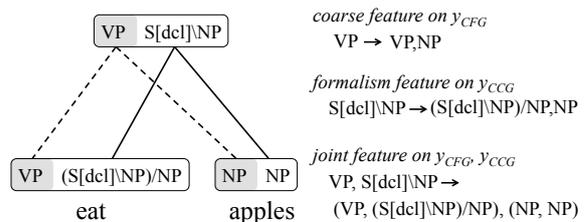


Figure 2: Illustration of the joint CCG-CFG representation. The shadowed labels correspond to the CFG derivation  $y_{CFG}$ , whereas the other labels correspond to the CCG derivation  $y_{CCG}$ . Note that the two derivations share the same (binarized) tree structure. Also shown are features that are turned on for this joint derivation (see Section 6).

how to generate CCG parses to unseen sentences.

## 4 A Joint Model for Two Formalisms

The key idea behind our work is to learn a joint distribution over CCG and CFG parses. Such a distribution can be marginalized to obtain a distribution over CCG or CFG and is thus appropriate when the training data is not parallel, as it is in our setting.

It is not immediately clear how to jointly model the CCG and CFG parses, which are structurally quite different. Furthermore, a joint distribution over these will become difficult to handle computationally if not constructed carefully. To address this difficulty, we make several simplifying assumptions. First, we assume that both parses are given in normal form, i.e., they correspond to binary derivation trees. CCG parses are already provided in this form in CCGBank. CFG parses in the Penn Treebank are not binary, and we therefore binarize them, as explained in Section 5.3.

Second, we assume that any  $y_{CFG}$  and  $y_{CCG}$  jointly generated must share the same derivation tree structure. This makes sense. Since both formalisms are constituency-based, their trees are expected to describe the same constituents. We denote the set of valid CFG and CCG joint parses for sentence  $S$  by  $\mathcal{Y}(S)$ .

The above two simplifying assumptions make it easy to define joint features on the two parses, as explained in Section 6. The representation and features are illustrated in Figure 2.

We shall work within the discriminative framework, where given a sentence we model a distribution over parses. As is standard in such settings, the distribution will be log-linear in a set of features of these parses. Denoting  $y = (y_{CFG}, y_{CCG})$ , we seek to model the distribution

$p(y|S)$  corresponding to the probability of generating a pair of parses (CFG and CCG) given a sentence. The distribution thus has the following form:

$$p_{joint}(y|S; \theta) = \frac{1}{Z(S; \theta)} e^{f(y,S) \cdot \theta}. \quad (1)$$

where  $\theta$  is a vector of parameters to be learned from data, and  $f(y, S)$  is a feature vector.  $Z(S; \theta)$  is a normalization (partition) function normalized over  $y \in \mathcal{Y}(S)$  the set of valid joint parses.

The feature vector contains three types of features: CFG specific, CCG specific and joint CFG-CCG. We denote these by  $f_{CFG}, f_{CCG}, f_{joint}$ . These depend on  $y_{CCG}, y_{CFG}$  and  $y$  respectively. Accordingly, the parameter vector  $\theta$  is a concatenation of  $\theta_{CCG}, \theta_{CFG}$  and  $\theta_{joint}$ .

As mentioned above, we can use Equation 1 to obtain distributions over  $y_{CCG}$  and  $y_{CFG}$  via marginalization. For the distribution over  $y_{CCG}$  we do precisely this, namely use:

$$p_{CCG}(y_{CCG}|S; \theta) = \sum_{y_{CFG}} p_{joint}(y|S; \theta) \quad (2)$$

For the distribution over  $y_{CFG}$  we could have marginalized  $p_{joint}$  over  $y_{CCG}$ . However, this computation is costly for each sentence, and has to be repeated for all the sentences. Instead, we assume that the distribution over  $y_{CFG}$  is a log-linear model with parameters  $\theta_{CFG}$  (i.e., a sub-vector of  $\theta$ ), namely:

$$p_{CFG}(y_{CFG}|S; \theta_{CFG}) = \frac{e^{f_{CFG}(y_{CFG}, S) \cdot \theta_{CFG}}}{Z(S; \theta_{CFG})}. \quad (3)$$

Thus, we assume that both  $p_{joint}$  and  $p_{CFG}$  have the same dependence on the  $f_{CFG}$  features.

**The Likelihood Objective:** Given the models above, it is natural to use maximum likelihood to find the optimal parameters. To do this, we define the following regularized likelihood function:

$$L(\theta) = \sum_{i=1}^N \log(p_{CFG}(y_{CFG}^i | S_i, \theta_{CFG})) + \sum_{i=1}^M \log(p_{CCG}(y_{CCG}^i | \bar{S}_i, \theta)) - \frac{\lambda}{2} \|\theta\|_2^2$$

where  $p_{CCG}$  and  $p_{CFG}$  are defined in Equations 2 and 3 respectively. The last term is the  $l_2$ -norm regularization. Our goal is then to find a  $\theta$  that maximizes  $L(\theta)$ .

**Training Algorithm:** For maximizing  $L(\theta)$  w.r.t.  $\theta$  we use the limited-memory BFGS algorithm (Nocedal and Wright, 1999). Calculating the gradient of  $L(\theta)$  requires evaluating the expected values of  $f(y, S)$  and  $f_{CFG}$  under the distributions  $p_{joint}$  and  $p_{CFG}$  respectively. This can be done via the inside-outside algorithm.<sup>3</sup>

**Parsing Using the Model:** To parse a sentence  $S$ , we calculate the maximum probability assignment for  $p_{joint}(y|S; \theta)$ .<sup>4</sup> The result is both a CFG and a CCG parse. Here we will mostly be interested in the CCG parse. The joint parse with maximum probability is found using a standard CYK chart parsing algorithm. The chart construction will be explained in Section 5.

## 5 Implementation

This section introduces important implementation details, including supertagging, feature forest pruning and binarization methods. Finally, we explain how to generalize our model to other constituency-based formalisms.

### 5.1 Supertagging

When parsing a target formalism tree, one needs to associate each word with a lexical entry. However, since the number of candidates is typically more than one thousand, the size of the chart explodes. One effective way of reducing the number of candidates is via supertagging (Clark and Curran, 2007). A supertagger is used for selecting a small set of lexical entry candidates for each word in the sentence. We use the tagger in (Clark and Curran, 2007) as a general supertagger for all the grammars considered. The only difference is that we use different lexical entries in different grammars.

### 5.2 Feature Forest Pruning

In the BFGS algorithm (see Section 4), feature expectation is computed using the inside-outside algorithm. To perform this dynamic programming efficiently, we first need to build the packed chart, namely the feature forest (Miyao, 2006) to represent the exponential number of all possible tree

<sup>3</sup>To speed up the implementation, gradient computation is parallelized, using the Message Passing Interface package (Gropp et al., 1999).

<sup>4</sup>An alternative approach would be to marginalize over  $y_{CFG}$  and maximize over  $y_{CCG}$ . However, this is a harder computational problem.

structures. However, a common problem for lexicalized grammars is that the forest size is too large. In CFG, the forest is pruned according to the inside probability of a simple generative PCFG model and a prior (Collins, 2003). The basic idea is to prune the trees with lower probability. For the target formalism, a common practice is to prune the forest using the supertagger (Clark and Curran, 2007; Miyao, 2006). In our implementation, we applied all pruning techniques, because the forest is a combination of CFG and target grammar formalisms (e.g., CCG or HPSG).

### 5.3 Binarization

We assume that the derivation tree in the target formalism is in a normal form, which is indeed the case for the treebanks we consider. As mentioned in Section 4, we would also like to work with binarized CFG derivations, such that all trees are in normal form and it is easy to construct features that link the two (see Section 6).

Since Penn Treebank trees are not binarized, we construct a simple procedure for binarizing them. The procedure is based on the available target formalism parses in the training corpus, which are binarized. We illustrate it with an example. In what follows, we describe derivations using the POS of the head words of the corresponding node in the tree. This makes it possible to transfer binarization rules between formalisms.

Suppose we want to learn the binarization rule of the following derivation in CFG:

$$NN \rightarrow (DT \ JJ \ NN) \quad (4)$$

We now look for binary derivations with these POS in the target formalism corpus, and take the most common binarization form. For example, we may find that the most common binarization to binarize the CFG derivation in Equation 4 is:

$$NN \rightarrow (DT \ (JJ \ NN))$$

If no (DT JJ NN) structure is observed in the CCG corpus, we first apply the binary branching on the children to the left of the head, and then on the children to the right of the head.

We also experiment with using fixed binarization rules such as left/right branching, instead of learning them. This results in a drop on the dependency F-score by about 5%.

### 5.4 Implementation in Other Formalisms

We introduce our model in the context of CCG, but the model can easily be generalized to other constituency-based grammars, such as HPSG and LFG. In a derivation tree, the formalism-specific information is mainly encoded in the lexical entries and the applied grammar rules, rather than the tree structures. Therefore we only need to change the node labels and lexical entries to the language-specific ones, while the framework of the model remains the same.

## 6 Features

Feature functions in log-linear models are designed to capture the characteristics of each derivation in the tree. In our model, as mentioned in Section 1, the features are also defined to enable information transfer between coarse and rich formalisms. In this section, we first introduce how different types of feature templates are designed, and then show an example of how the features help transfer the syntactic structure information. Note that the same feature templates are used for all the target grammar formalisms.

Recall that our  $y$  contains both the CFG and CCG parses, and that these use the same derivation tree structure. Each feature will consider either the CFG derivation, the CCG derivation or these two derivations jointly.

The feature construction is similar to constructions used in previous work (Miyao, 2006). The features are based on the atomic features listed in Table 1. These will be used to construct  $f(y, S)$  as explained next.

$hl$	lexical entries/CCG categories of the head word
$r$	grammar rules, i.e. HPSG schema, resulting CCG categories, LFG mapping equations
$sy$	CFG syntactic label of the node (e.g. NP, VP)
$d$	distance between the head words of the children
$c$	whether a comma exists between the head words of the children
$sp$	the span of the subtree rooted at the node
$hw$	surface form of the head word of the node
$hp$	part-of-speech of the head word
$p_i$	part-of-speech of the $i$ -th word in the sentence

Table 1: Templates of atomic features.

We define the following feature templates:  $f_{binary}$  for binary derivations,  $f_{unary}$  for unary derivations, and  $f_{root}$  for the root nodes. These use the atomic features in Table 1, resulting in the



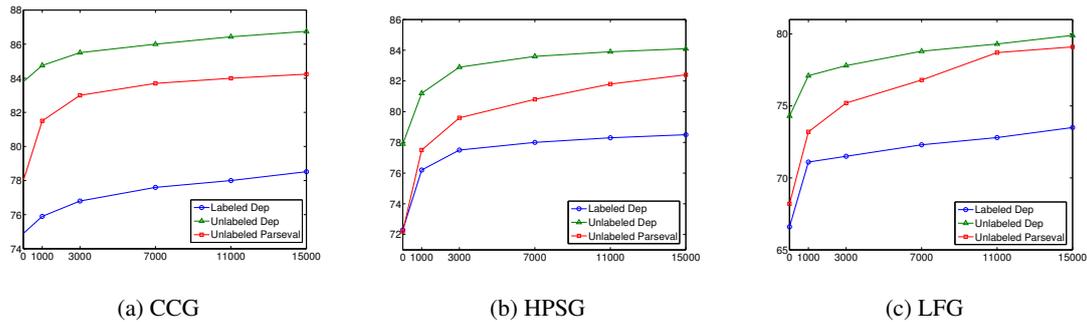


Figure 4: Model performance with 500 target formalism trees and different numbers of CFG trees, evaluated using labeled/unlabeled dependency F-score and unlabeled PARSEVAL.

crafted rules (Hockenmaier and Steedman, 2002; Miyao et al., 2005). Table 3 shows which sections of the treebanks were used in training, testing and development for both formalisms. Our LFG training dataset was constructed in a similar fashion (Cahill et al., 2002). However, we choose to use PARC700 as our LFG testing and development datasets, following the previous work by (Kaplan et al., 2004). It contains 700 manually annotated sentences that are randomly selected from Penn Treebank Section 23. The split of PARC700 follows the setting in (Kaplan et al., 2004). Since our model does not assume parallel data, we use distinct sentences in the source and target treebanks. Following previous work (Hockenmaier, 2003; Miyao and Tsujii, 2008), we only consider sentences not exceeding 40 words, except on PARC700 where all sentences are used.

**Evaluation Metrics:** We use two evaluation metrics. First, following previous work, we evaluate our method using the labeled and unlabeled predicate-argument dependency F-score. This metric is commonly used to measure parsing quality for the formalisms considered in this paper. The detailed definition of this measure as applied for each formalism is provided in (Clark and Curran, 2003; Miyao and Tsujii, 2008; Cahill et al., 2004). For CCG, we use the evaluation script from the C&C tools.<sup>5</sup> For HPSG, we evaluate all types of dependencies, including punctuations. For LFG, we consider the *preds-only* dependencies, which are the dependencies between pairs of words. Secondly, we also evaluate using unlabeled PARSEVAL, a standard measure for PCFG parsing (Petrov and Klein, 2007; Charniak and Johnson, 2005; Charniak, 2000; Collins, 1997). The dependency F-score captures both the target-

grammar labels and tree-structural relations. The unlabeled PARSEVAL is used as an auxiliary measure that enables us to separate these two aspects by focusing on the structural relations exclusively.

**Training without CFG Data:** To assess the impact of coarse data in the experiments below, we also consider the model trained only on formalism-specific annotations. When no CFG sentences are available, we assign all the CFG labels to a special value shared by all the nodes. In this set-up, the model reduces to a normal log-linear model for the target formalism.

**Parameter Settings:** During training, all the feature parameters  $\theta$  are initialized to zero. The hyperparameters used in the model are tuned on the development sets. We noticed, however, that the resulting values are consistent across different formalisms. In particular, we set the  $l_2$ -norm weight to  $\lambda = 1.0$ , the supertagger threshold to  $\beta = 0.01$ , and the PCFG pruning threshold to  $\alpha = 0.002$ .

## 8 Experiment and Analysis

**Impact of Coarse Annotations on Target Formalism:** To analyze the effectiveness of annotation transfer, we fix the number of annotated trees in the target formalism and vary the amount of coarse annotations available to the algorithm during training. In particular, we use 500 sentences with formalism-specific annotations, and vary the number of CFG trees from zero to 15,000.

As Figure 4 shows, CFG data boosts parsing accuracy for all the target formalisms. For instance, there is a gain of 6.2% in labeled dependency F-score for HPSG formalism when 15,000 CFG trees are used. Moreover, increasing the number of coarse annotations used in training leads to further improvement on different evaluation metrics.

<sup>5</sup> Available at <http://svn.ask.it.usyd.edu.au/trac/candc/wiki>

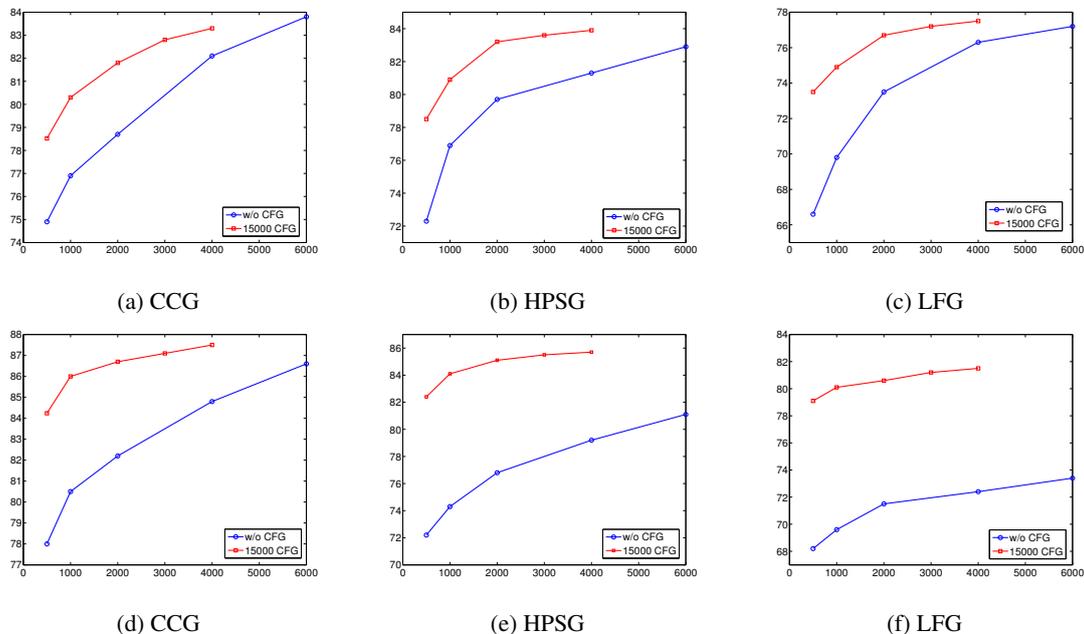


Figure 5: Model performance with different target formalism trees and zero or 15,000 CFG trees. The first row shows the results of labeled dependency F-score and the second row shows the results of unlabeled PARSEVAL.

### Tradeoff between Target and Coarse Annotations:

We also assess the relative contribution of coarse annotations when the size of annotated training corpus in the target formalism varies. In this set of experiments, we fix the number of CFG trees to 15,000 and vary the number of target annotations from 500 to 4,000. Figure 5 shows the relative contribution of formalism-specific annotations compared to that of the coarse annotations. For instance, Figure 5a shows that the parsing performance achieved using 2,000 CCG sentences can be achieved using approximately 500 CCG sentences when coarse annotations are available for training. More generally, the result convincingly demonstrates that coarse annotations are helpful for all the sizes of formalism-specific training data. As expected, the improvement margin decreases when more formalism-specific data is used.

Figure 5 also illustrates a slightly different characteristics of transfer performance between two evaluation metrics. Across all three grammars, we can observe that adding CFG data has a more pronounced effect on the PARSEVAL measure than the dependency F-score. This phenomenon can be explained as follows. The unlabeled PARSEVAL score (Figure 5d-f) mainly relies on the coarse structural information. On the other hand, predicate-argument dependency F-score (Figure 5a-c) also relies on the target grammar information. Because that our model only transfers structural information from the source

treebank, the gains of PARSEVAL are expected to be larger than that of dependency F-score.

Grammar	Parser	# Grammar trees	
		1,000	15,000
CCG	C&C	74.1 / 83.4	82.6 / 90.1
	Model	76.8 / 85.5	84.7 / 90.9
HPSG	Enju	75.8 / 80.6	84.2 / 87.3
	Model	76.9 / 82.0	84.9 / 88.3
LFG	Pipeline Annotator	68.5 / 74.0	82.6 / 85.9
	Model	69.8 / 76.6	81.1 / 84.7

Table 4: The labeled/unlabeled dependency F-score comparisons between our model and state-of-the-art parsers.

**Comparison to State-of-the-art Parsers:** We would also like to demonstrate that the above gains of our transfer model are achieved using an adequate formalism-specific parser. Since our model can be trained exclusively on formalism-specific data, we can compare it to state-of-the-art formalism-specific parsers. For this experiment, we choose the C&C parser (Clark and Curran, 2003) for CCG, Enju parser (Miyao and Tsujii, 2008) for HPSG and pipeline automatic annotator (Cahill et al., 2004) with Charniak parser for LFG. For all three parsers, we use the implementation provided by the authors with the default parameter values. All the models are trained on either 1,000 or 15,000 sentences annotated with formalism-specific trees, thus evaluating their performances on small scale or large scale of data. As Table 4 shows, our model is competitive with

all the baselines described above. It’s not surprising that Cahill’s model outperforms our log-linear model because it relies heavily on hand-crafted rules optimized for the dataset.

**Correspondence between CFG and Target Formalisms:** Finally, we analyze highly weighted features. Table 5 shows such features for HPSG; similar patterns are also found for the other grammar formalisms. The first two features are formalism-specific ones, the first for HPSG and the second for CFG. They show that we correctly learn a frequent derivation in the target formalism and CFG. The third one shows an example of a connection between CFG and the target formalism. Our model correctly learns that a syntactic derivation with children VP and NP is very likely to be mapped to the derivation  $(\text{head\_comp}) \rightarrow ([N\langle V \rangle N], [N.3\text{sg}])$  in HPSG.

Feature type	Features with high weight
Target formalism	<i>Template</i> $(r) \rightarrow (hl_l, hp_l)(hl_r, pr)$ <i>Examples</i> $(\text{head\_comp}) \rightarrow ([N\langle V \rangle N], \text{VB})([N.3\text{sg}], \text{NN})$
Coarse formalism	<i>Template</i> $(sy_p) \rightarrow (sy_l, hp_l)(sy_r, hp_r)$ <i>Examples</i> $(\text{VP}) \rightarrow (\text{VP}, \text{VB})(\text{NP}, \text{NN})$
Joint features	<i>Template</i> $(r) \rightarrow (hl_l, sy_l)(le_r, sy_r)$ <i>Examples</i> $(\text{head\_comp}) \rightarrow ([N\langle V \rangle N], \text{VP})([N.3\text{sg}], \text{NP})$

Table 5: Example features with high weight.

## 9 Conclusions

We present a method for cross-formalism transfer in parsing. Our model utilizes coarse syntactic annotations to supplement a small number of formalism-specific trees for training on constituency-based grammars. Our experimental results show that across a range of such formalisms, the model significantly benefits from the coarse annotations.

## Acknowledgments

The authors acknowledge the support of the Army Research Office (grant 1130128-258552). We thank Yusuke Miyao, Ozlem Cetinoglu, Stephen Clark, Michael Auli and Yue Zhang for answering questions and sharing the codes of their work. We also thank the members of the MIT NLP group

and the ACL reviewers for their suggestions and comments. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors, and do not necessarily reflect the views of the funding organizations.

## References

- John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128. Association for Computational Linguistics.
- Joan Bresnan. 1982. *The mental representation of grammatical relations*, volume 1. The MIT Press.
- Aoife Cahill, Mairad McCarthy, Josef van Genabith, and Andy Way. 2002. Parsing with pcfgs and automatic f-structure annotation. In *Proceedings of the Seventh International Conference on LFG*, pages 76–95. CSLI Publications.
- Aoife Cahill, Michael Burke, Ruth O’Donovan, Josef Van Genabith, and Andy Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage pcfg-based lfg approximations. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 319. Association for Computational Linguistics.
- Aoife Cahill. 2004. *Parsing with Automatically Acquired, Wide-Coverage, Robust, Probabilistic LFG Approximation*. Ph.D. thesis.
- Marie Candito, Benoît Crabbé, Pascal Denis, et al. 2010. Statistical french dependency parsing: tree-bank conversion and first results. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, pages 1840–1847.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 173–180. Association for Computational Linguistics.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 132–139.
- John Chen and Vijay K Shanker. 2005. Automated extraction of tags from the penn treebank. *New developments in parsing technology*, pages 73–89.
- Stephen Clark and James R Curran. 2003. Log-linear models for wide-coverage ccg parsing. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 97–104. Association for Computational Linguistics.

- Stephen Clark and James R Curran. 2007. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552.
- Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 16–23. Association for Computational Linguistics.
- Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational linguistics*, 29(4):589–637.
- Mark Dredze, John Blitzer, Partha Pratim Talukdar, Kuzman Ganchev, Joao V Graça, and Fernando Pereira. 2007. Frustratingly hard domain adaptation for dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, volume 2007.
- William Gropp, Ewing Lusk, and Anthony Skjellum. 1999. *Using MPI: portable parallel programming with the message passing interface*, volume 1. MIT press.
- Julia Hockenmaier and Mark Steedman. 2002. Acquiring compact lexicalized grammars from a cleaner treebank. In *Proceedings of the Third LREC Conference*, pages 1974–1981.
- Julia Hockenmaier. 2003. Data and models for statistical parsing with combinatory categorial grammar.
- Rebecca Hwa, Philip Resnik, and Amy Weinberg. 2005. Breaking the resource bottleneck for multilingual parsing. Technical report, DTIC Document.
- Wenbin Jiang and Qun Liu. 2009. Automatic adaptation of annotation standards for dependency parsing: using projected treebank as source corpus. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 25–28. Association for Computational Linguistics.
- Ronald M. Kaplan, Stefan Riezler, Tracy H. King, John T. Maxwell III, Alexander Vasserman, and Richard Crouch. 2004. Speed and accuracy in shallow and deep stochastic parsing. In *Proceedings of NAACL*.
- Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M Kaplan. 2003. The parc 700 dependency bank. In *Proceedings of the EACL03: 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*, pages 1–8.
- Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.
- David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic domain adaptation for parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 28–36. Association for Computational Linguistics.
- Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 216–220. Association for Computational Linguistics.
- Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 62–72. Association for Computational Linguistics.
- Yusuke Miyao and Jun’ichi Tsujii. 2008. Feature forest models for probabilistic hpsg parsing. *Computational Linguistics*, 34(1):35–80.
- Yusuke Miyao, Takashi Ninomiya, and Junichi Tsujii. 2005. Corpus-oriented grammar development for acquiring a head-driven phrase structure grammar from the penn treebank. *Natural Language Processing–IJCNLP 2004*, pages 684–693.
- Yusuke Miyao. 2006. *From Linguistic Theory to Syntactic Analysis: Corpus-Oriented Grammar Development and Feature Forest Model*. Ph.D. thesis.
- Jorge Nocedal and Stephen J Wright. 1999. *Numerical optimization*. Springer verlag.
- Stephan Oepen, Dan Flickinger, and Francis Bond. 2004. Towards holistic grammar engineering and testing-grafting treebank maintenance into the grammar revision cycle. In *Proceedings of the IJCNLP workshop beyond shallow analysis*. Citeseer.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 404–411.
- Carl Pollard and Ivan A Sag. 1994. *Head-driven phrase structure grammar*. University of Chicago Press.
- Stefan Riezler, Tracy H King, Ronald M Kaplan, Richard Crouch, John T Maxwell III, and Mark Johnson. 2002. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 271–278. Association for Computational Linguistics.
- Benjamin Snyder, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the*

*4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 73–81. Association for Computational Linguistics.

Mark Steedman. 2001. *The syntactic process*. MIT press.

Yue Zhang, Stephen Clark, et al. 2011. Shift-reduce ccg parsing. In *Proceedings of the 49th Meeting of the Association for Computational Linguistics*, pages 683–692.

# A Context Free TAG Variant

**Ben Swanson**

Brown University  
Providence, RI  
chonger@cs.brown.edu

**Elif Yamangil**

Harvard University  
Cambridge, MA  
elif@eecs.harvard.edu

**Eugene Charniak**

Brown University  
Providence, RI  
ec@cs.brown.edu

**Stuart Shieber**

Harvard University  
Cambridge, MA  
shieber@eecs.harvard.edu

## Abstract

We propose a new variant of Tree-Adjoining Grammar that allows adjunction of full wrapping trees but still bears only context-free expressivity. We provide a transformation to context-free form, and a further reduction in probabilistic model size through factorization and pooling of parameters. This collapsed context-free form is used to implement efficient grammar estimation and parsing algorithms. We perform parsing experiments the Penn Treebank and draw comparisons to Tree-Substitution Grammars and between different variations in probabilistic model design. Examination of the most probable derivations reveals examples of the linguistically relevant structure that our variant makes possible.

## 1 Introduction

While it is widely accepted that natural language is not context-free, practical limitations of existing algorithms motivate Context-Free Grammars (CFGs) as a good balance between modeling power and asymptotic performance (Charniak, 1996). In constituent-based parsing work, the prevailing technique to combat this divide between efficient models and real world data has been to selectively strengthen the dependencies in a CFG by increasing the grammar size through methods such as symbol refinement (Petrov et al., 2006).

Another approach is to employ a more powerful grammatical formalism and devise constraints and transformations that allow use of essential efficient algorithms such as the Inside-Outside algorithm (Lari and Young, 1990) and CYK parsing. Tree-Adjoining Grammar (TAG) is a natural

starting point for such methods as it is the canonical member of the mildly context-sensitive family, falling just above CFGs in the hierarchy of formal grammars. TAG has a crucial advantage over CFGs in its ability to represent long distance interactions in the face of the interposing variations that commonly manifest in natural language (Joshi and Schabes, 1997). Consider, for example, the sentences

These pretzels are making me thirsty.  
These pretzels are not making me thirsty.  
These pretzels that I ate are making me thirsty.

Using a context-free language model with proper phrase bracketing, the connection between the words *pretzels* and *thirsty* must be recorded with three separate patterns, which can lead to poor generalizability and unreliable sparse frequency estimates in probabilistic models. While these problems can be overcome to some extent with large amounts of data, redundant representation of patterns is particularly undesirable for systems that seek to extract coherent and concise information from text.

TAG allows a linguistically motivated treatment of the example sentences above by generating the last two sentences through modification of the first, applying operations corresponding to negation and the use of a subordinate clause. Unfortunately, the added expressive power of TAG comes with  $O(n^6)$  time complexity for essential algorithms on sentences of length  $n$ , as opposed to  $O(n^3)$  for the CFG (Schabes, 1990). This makes TAG infeasible to analyze real world data in a reasonable time frame.

In this paper, we define oSTAG, a new way to constrain TAG in a conceptually simple way so

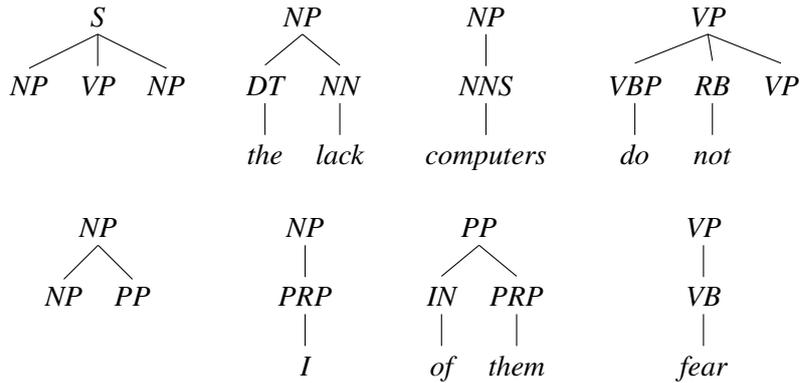


Figure 1: A simple Tree-Substitution Grammar using  $S$  as its start symbol. This grammar derives the sentences from a quote of Isaac Asimov’s - “I do not fear computers. I fear the lack of them.”

that it can be reduced to a CFG, allowing the use of traditional cubic-time algorithms. The reduction is reversible, so that the original TAG derivation can be recovered exactly from the CFG parse. We provide this reduction in detail below and highlight the compression afforded by this TAG variant on synthetic formal languages.

We evaluate OSTAG on the familiar task of parsing the Penn Treebank. Using an automatically induced Tree-Substitution Grammar (TSG), we heuristically extract an OSTAG and estimate its parameters from data using models with various reduced probabilistic models of adjunction. We contrast these models and investigate the use of adjunction in the most probable derivations of the test corpus, demonstrating the superior modeling performance of OSTAG over TSG.

## 2 TAG and Variants

Here we provide a short history of the relevant work in related grammar formalisms, leading up to a definition of OSTAG. We start with context-free grammars, the components of which are  $\langle N, T, R, S \rangle$ , where  $N$  and  $T$  are the sets of non-terminal and terminal symbols respectively, and  $S$  is a distinguished nonterminal, the start symbol. The rules  $R$  can be thought of as elementary trees of depth 1, which are combined by substituting a derived tree rooted at a nonterminal  $X$  at some leaf node in an elementary tree with a frontier node labeled with that same nonterminal. The derived trees rooted at the start symbol  $S$  are taken to be the trees generated by the grammar.

### 2.1 Tree-Substitution Grammar

By generalizing CFG to allow elementary trees in  $R$  to be of depth greater than or equal to 1, we

get the Tree-Substitution Grammar. TSG remains in the family of context-free grammars, as can be easily seen by the removal of the internal nodes in all elementary trees; what is left is a CFG that generates the same language. As a reversible alternative that preserves the internal structure, annotation of each internal node with a unique index creates a large number of deterministic CFG rules that record the structure of the original elementary trees. A more compact CFG representation can be obtained by marking each node in each elementary tree with a signature of its subtree. This transform, presented by Goodman (2003), can rein in the grammar constant  $G$ , as the crucial CFG algorithms for a sentence of length  $n$  have complexity  $O(Gn^3)$ .

A simple probabilistic model for a TSG is a set of multinomials, one for each nonterminal in  $N$  corresponding to its possible substitutions in  $R$ . A more flexible model allows a potentially infinite number of substitution rules using a Dirichlet Process (Cohn et al., 2009; Cohn and Blunsom, 2010). This model has proven effective for grammar induction via Markov Chain Monte Carlo (MCMC), in which TSG derivations of the training set are repeatedly sampled to find frequently occurring elementary trees. A straightforward technique for induction of a finite TSG is to perform this non-parametric induction and select the set of rules that appear in at least one sampled derivation at one or several of the final iterations.

### 2.2 Tree-Adjoining Grammar

Tree-adjoining grammar (TAG) (Joshi, 1985; Joshi, 1987; Joshi and Schabes, 1997) is an extension of TSG defined by a tuple  $\langle N, T, R, A, S \rangle$ , and differs from TSG only in the addition of a

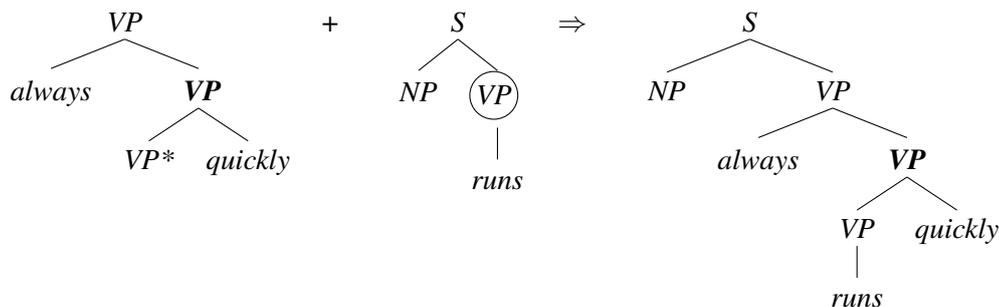


Figure 2: The adjunction operation combines the auxiliary tree (left) with the elementary tree (middle) to form a new derivation (right). The adjunction site is circled, and the foot node of the auxiliary tree is denoted with an asterisk. The OSTAG constraint would disallow further adjunction at the bold **VP** node only, as it is along the spine of the auxiliary tree.

set of auxiliary trees  $A$  and the adjunction operation that governs their use. An auxiliary tree  $\alpha$  is an elementary tree containing a single distinguished nonterminal leaf, the foot node, with the same symbol as the root of  $\alpha$ . An auxiliary tree with root and foot node  $X$  can be adjoined into an internal node of an elementary tree labeled with  $X$  by splicing the auxiliary tree in at that internal node, as pictured in Figure 2. We refer to the path between the root and foot nodes in an auxiliary tree as the *spine* of the tree.

As mentioned above, the added power afforded by adjunction comes at a serious price in time complexity. As such, probabilistic modeling for TAG in its original form is uncommon. However, a large effort in non-probabilistic grammar induction has been performed through manual annotation with the XTAG project (Doran et al., 1994).

### 2.3 Tree Insertion Grammar

Tree Insertion Grammars (TIGs) are a longstanding compromise between the intuitive expressivity of TAG and the algorithmic simplicity of CFGs. Schabes and Waters (1995) showed that by restricting the form of the auxiliary trees in  $A$  and the set of auxiliary trees that may adjoin at particular nodes, a TAG generates only context-free languages. The TIG restriction on auxiliary trees states that the foot node must occur as either the leftmost or rightmost leaf node. This introduces an important distinction between left, right, and wrapping auxiliary trees, of which only the first two are allowed in TIG. Furthermore, TIG disallows adjunction of left auxiliary trees on the spines of right auxiliary trees, and vice versa. This is to prevent the construction of wrapping auxiliary trees, whose removal is essential for the simplified

complexity of TIG.

Several probabilistic models have been proposed for TIG. While earlier approaches such as Hwa (1998) and Chiang (2000) relied on heuristic induction methods, they were nevertheless successful at parsing. Later approaches (Shindo et al., 2011; Yamangil and Shieber, 2012) were able to extend the non-parametric modeling of TSGs to TIG, providing methods for both modeling and grammar induction.

### 2.4 OSTAG

Our new TAG variant is extremely simple. We allow arbitrary initial and auxiliary trees, and place only one restriction on adjunction: we disallow adjunction at any node on the spine of an auxiliary tree below the root (though we discuss relaxing that constraint in Section 4.2). We refer to this variant as Off Spine TAG (OSTAG) and note that it allows the use of full wrapping rules, which are forbidden in TIG. This targeted blocking of recursion has similar motivations and benefits to the approximation of CFGs with regular languages (Mohri and Jan Nederhof, 2000).

The following sections discuss in detail the context-free nature of OSTAG and alternative probabilistic models for its equivalent CFG form. We propose a simple but empirically effective heuristic for grammar induction for our experiments on Penn Treebank data.

## 3 Transformation to CFG

To demonstrate that OSTAG has only context-free power, we provide a reduction to context-free grammar. Given an OSTAG  $\langle N, T, R, A, S \rangle$ , we define the set  $\mathcal{N}$  of nodes of the corresponding CFG to be pairs of a tree in  $R$  or  $A$  together with an

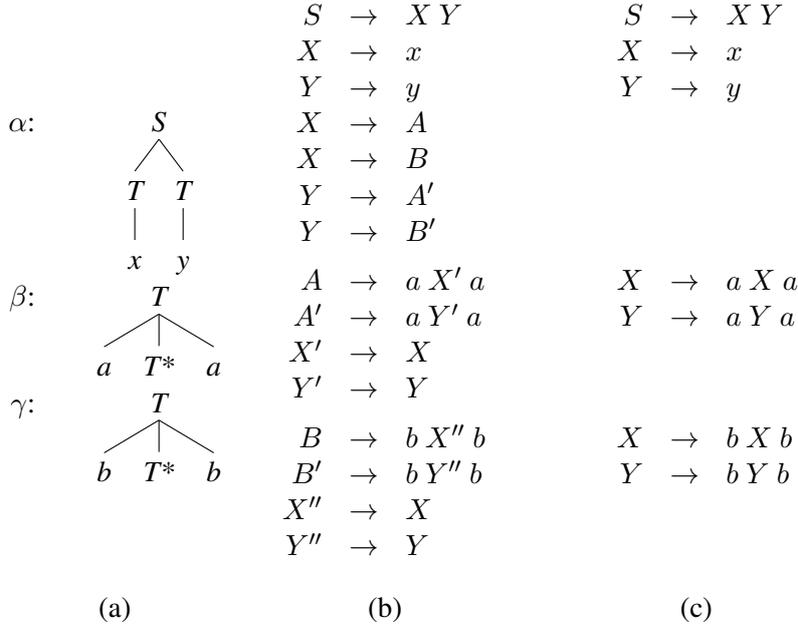


Figure 3: (a) OSTAG for the language  $xxw^Ryyv^R$  where  $w, v \in \{a|b\}^+$  and  $R$  reverses a string. (b) A CFG for the same language, which of necessity must distinguish between nonterminals  $X$  and  $Y$  playing the role of  $T$  in the OSTAG. (c) Simplified CFG, conflating nonterminals, but which must still distinguish between  $X$  and  $Y$ .

address (Gorn number (Gorn, 1965)) in that tree. We take the nonterminals of the target CFG grammar to be nodes or pairs of nodes, elements of the set  $\mathcal{N} + \mathcal{N} \times \mathcal{N}$ . We notate the pairs of nodes with a kind of “applicative” notation. Given two nodes  $\eta$  and  $\eta'$ , we notate a target nonterminal as  $\eta(\eta')$ .

Now for each tree  $\tau$  and each interior node  $\eta$  in  $\tau$  that is not on the spine of  $\tau$ , with children  $\eta_1, \dots, \eta_k$ , we add a context-free rule to the grammar

$$\eta \rightarrow \eta_1 \cdots \eta_k \quad (1)$$

and if interior node  $\eta$  is on the spine of  $\tau$  with  $\eta_s$  the child node also on the spine of  $\tau$  (that is, dominating the foot node of  $\tau$ ) and  $\eta'$  is a node (in any tree) where  $\tau$  is adjoinable, we add a rule

$$\eta(\eta') \rightarrow \eta_1 \cdots \eta_s(\eta') \cdots \eta_k \quad . \quad (2)$$

Rules of type (1) handle the expansion of a node not on the spine of an auxiliary tree and rules of type (2) a spinal node.

In addition, to initiate adjunction at any node  $\eta'$  where a tree  $\tau$  with root  $\eta$  is adjoinable, we use a rule

$$\eta' \rightarrow \eta(\eta') \quad (3)$$

and for the foot node  $\eta_f$  of  $\tau$ , we use a rule

$$\eta_f(\eta) \rightarrow \eta \quad (4)$$

The OSTAG constraint follows immediately from the structure of the rules of type (2). Any child spine node  $\eta_s$  manifests as a CFG nonterminal  $\eta_s(\eta')$ . If child spine nodes themselves allowed adjunction, we would need a type (3) rule of the form  $\eta_s(\eta') \rightarrow \eta_s(\eta')(\eta'')$ . This rule itself would feed adjunction, requiring further stacking of nodes, and an infinite set of CFG nonterminals and rules. This echoes exactly the stacking found in the LIG reduction of TAG.

To handle substitution, any frontier node  $\eta$  that allows substitution of a tree rooted with node  $\eta'$  engenders a rule

$$\eta \rightarrow \eta' \quad (5)$$

This transformation is reversible, which is to say that each parse tree derived with this CFG implies exactly one OSTAG derivation, with substitutions and adjunctions coded by rules of type (5) and (3) respectively. Depending on the definition of a TAG derivation, however, the converse is not necessarily true. This arises from the spurious ambiguity between adjunction at a substitution site (before applying a type (5) rule) versus the same adjunction at the root of the substituted initial tree (after applying a type (5) rule). These choices lead to different derivations in CFG form, but their TAG derivations can be considered conceptually

identical. To avoid double-counting derivations, which can adversely effect probabilistic modeling, type (3) and type (4) rules in which the side with the unapplied symbol is a nonterminal leaf can be omitted.

### 3.1 Example

The grammar of Figure 3(a) can be converted to a CFG by this method. We indicate for each CFG rule its type as defined above the production arrow. All types are used save type (5), as substitution is not employed in this example. For the initial tree  $\alpha$ , we have the following generated rules (with nodes notated by the tree name and a Gorn number subscript):

$$\begin{array}{ll} \alpha_\epsilon \xrightarrow{1} \alpha_1 \alpha_2 & \alpha_1 \xrightarrow{3} \beta_\epsilon(\alpha_1) \\ \alpha_1 \xrightarrow{1} x & \alpha_1 \xrightarrow{3} \gamma_\epsilon(\alpha_1) \\ \alpha_2 \xrightarrow{1} y & \alpha_2 \xrightarrow{3} \beta_\epsilon(\alpha_2) \\ & \alpha_2 \xrightarrow{3} \gamma_\epsilon(\alpha_2) \end{array}$$

For the auxiliary trees  $\beta$  and  $\gamma$  we have:

$$\begin{array}{ll} \beta_\epsilon(\alpha_1) \xrightarrow{2} a \beta_1(\alpha_1) a & \\ \beta_\epsilon(\alpha_2) \xrightarrow{2} a \beta_1(\alpha_2) a & \\ \beta_1(\alpha_1) \xrightarrow{4} \alpha_1 & \\ \beta_1(\alpha_2) \xrightarrow{4} \alpha_2 & \\ \gamma_\epsilon(\alpha_1) \xrightarrow{2} b \gamma_1(\alpha_1) b & \\ \gamma_\epsilon(\alpha_2) \xrightarrow{2} b \gamma_1(\alpha_2) b & \\ \gamma_1(\alpha_1) \xrightarrow{4} \alpha_1 & \\ \gamma_1(\alpha_2) \xrightarrow{4} \alpha_2 & \end{array}$$

The grammar of Figure 3(b) is simply a renaming of this grammar.

## 4 Applications

### 4.1 Compact grammars

The OSTAG framework provides some leverage in expressing particular context-free languages more compactly than a CFG or even a TSG can. As an example, consider the language of bracketed palindromes

$$Pal = a_i w a_i w^R a_i \mid \begin{array}{l} 1 \leq i \leq k \\ w \in \{b_j \mid 1 \leq j \leq m\}^* \end{array}$$

containing strings like  $a_2 b_1 b_3 a_2 b_3 b_1 a_2$ . Any TSG for this language must include as substrings some subpalindrome constituents for long enough strings. Whatever nonterminal covers such a

string, it must be specific to the  $a$  index within it, and must introduce at least one pair of  $b$ s as well. Thus, there are at least  $m$  such nonterminals, each introducing at least  $k$  rules, requiring at least  $km$  rules overall. The simplest such grammar, expressed as a CFG, is in Figure 4(a). The ability to use adjunction allows expression of the same language as an OSTAG with  $k + m$  elementary trees (Figure 4(b)). This example shows that an OSTAG can be quadratically smaller than the corresponding TSG or CFG.

### 4.2 Extensions

The technique in OSTAG can be extended to expand its expressiveness without increasing generative capacity.

First, OSTAG allows zero adjunctions on each node on the spine below the root of an auxiliary tree, but any non-zero finite bound on the number of adjunctions allowed on-spine would similarly limit generative capacity. The tradeoff is in the grammar constant of the effective probabilistic CFG; an extension that allows  $k$  levels of on spine adjunction has a grammar constant that is  $O(|\mathcal{N}|^{(k+2)})$ .

Second, the OSTAG form of adjunction is consistent with the TIG form. That is, we can extend OSTAG by allowing on-spine adjunction of left- or right-auxiliary trees in keeping with the TIG constraints without increasing generative capacity.

### 4.3 Probabilistic OSTAG

One major motivation for adherence to a context-free grammar formalism is the ability to employ algorithms designed for probabilistic CFGs such as the CYK algorithm for parsing or the Inside-Outside algorithm for grammar estimation. In this section we present a probabilistic model for an OSTAG grammar in PCFG form that can be used in such algorithms, and show that many parameters of this PCFG can be pooled or set equal to one and ignored. References to rules of types (1-5) below refer to the CFG transformation rules defined in Section 3. While in the preceding discussion we used Gorn numbers for clarity, our discussion applies equally well for the Goodman transform discussed above, in which each node is labeled with a signature of its subtree. This simply redefines  $\eta$  in the CFG reduction described in Section 3 to be a subtree indicator, and dramatically reduces redundancy in the generated grammar.

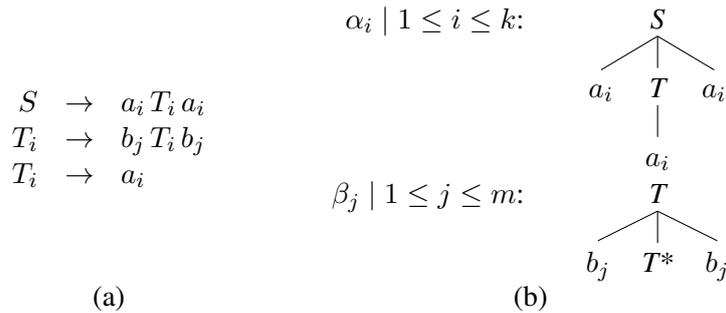


Figure 4: A CFG (a) and more compact OSTAG (b) for the language *Pal*

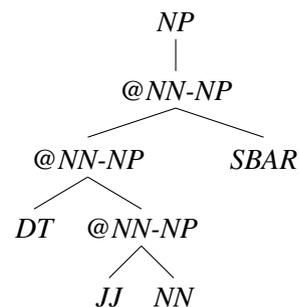
The first practical consideration is that CFG rules of type (2) are deterministic, and as such we need only record the rule itself and no associated parameter. Furthermore, these rules employ a template in which the stored symbol appears in the left-hand side and in exactly one symbol on the right-hand side where the spine of the auxiliary tree proceeds. One deterministic rule exists for this template applied to each  $\eta$ , and so we may record only the template. In order to perform CYK or IO, it is not even necessary to record the index in the right-hand side where the spine continues; these algorithms fill a chart bottom up and we can simply propagate the stored nonterminal up in the chart.

CFG rules of type (4) are also deterministic and do not require parameters. In these cases it is not necessary to record the rules, as they all have exactly the same form. All that is required is a check that a given symbol is adjoinable, which is true for all symbols except nonterminal leaves and applied symbols. Rules of type (5) are necessary to capture the probability of substitution and so we will require a parameter for each.

At first glance, it would seem that due to the identical domain of the left-hand sides of rules of types (1) and (3) a parameter is required for each such rule. To avoid this we propose the following factorization for the probabilistic expansion of an off spine node. First, a decision is made as to whether a type (1) or (3) rule will be used; this corresponds to deciding if adjunction will or will not take place at the node. If adjunction is rejected, then there is only one type (1) rule available, and so parameterization of type (1) rules is unnecessary. If we decide on adjunction, one of the available type (3) rules is chosen from a multinomial. By conditioning the probability of adjunction on varying amounts of information about the node, alternative models can easily be defined.

## 5 Experiments

As a proof of concept, we investigate OSTAG in the context of the classic Penn Treebank statistical parsing setup; training on section 2-21 and testing on section 23. For preprocessing, words that occur only once in the training data are mapped to the unknown categories employed in the parser of Petrov et al. (2006). We also applied the annotation from Klein and Manning (2003) that appends “-U” to each nonterminal node with a single child, drastically reducing the presence of looping unary chains. This allows the use of a coarse to fine parsing strategy (Charniak et al., 2006) in which a sentence is first parsed with the Maximum Likelihood PCFG and only constituents whose probability exceeds a cutoff of  $10^{-4}$  are allowed in the OSTAG chart. Designed to facilitate sister adjunction, we define our binarization scheme by example in which the added nodes, indicated by @, record both the parent and head child of the rule.



A compact TSG can be obtained automatically using the MCMC grammar induction technique of Cohn and Blunsom (2010), retaining all TSG rules that appear in at least one derivation in after 1000 iterations of sampling. We use EM to estimate the parameters of this grammar on sections 2-21, and use this as our baseline.

To generate a set of TAG rules, we consider each rule in our baseline TSG and find all possi-

	All	40	#Adj	#Wrap
TSG	85.00	86.08	–	–
TSG'	85.12	86.21	–	–
OSTAG <sup>1</sup>	85.42	86.43	1336	52
OSTAG <sup>2</sup>	85.54	86.56	1952	44
OSTAG <sup>3</sup>	85.86	86.84	3585	41

Figure 5: Parsing F-Score for the models under comparison for both the full test set and sentences of length 40 or less. For the OSTAG models, we list the number of adjunctions in the MPD of the full test set, as well as the number of wrapping adjunctions.

ble auxiliary root and foot node pairs it contains. For each such root/foot pair, we include the TAG rule implied by removal of the structure above the root and below the foot. We also include the TSG rule left behind when the adjunction of this auxiliary tree is removed. To be sure that experimental gains are not due to this increased number of TSG initial trees, we calculate parameters using EM for this expanded TSG and use it as a second baseline (TSG'). With our full set of initial and auxiliary trees, we use EM and the PCFG reduction described above to estimate the parameters of an OSTAG.

We investigate three models for the probability of adjunction at a node. The first uses a conservative number of parameters, with a Bernoulli variable for each symbol (OSTAG<sup>1</sup>). The second employs more parameters, conditioning on both the node's symbol and the symbol of its leftmost child (OSTAG<sup>2</sup>). The third is highly parameterized but most prone to data sparsity, with a separate Bernoulli distribution for each Goodman index  $\eta$  (OSTAG<sup>3</sup>). We report results for Most Probable Derivation (MPD) parses of section 23 in Figure 5.

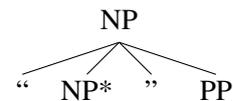
Our results show that OSTAG outperforms both baselines. Furthermore, the various parameterizations of adjunction with OSTAG indicate that, at least in the case of the Penn Treebank, the finer grained modeling of a full table of adjunction probabilities for each Goodman index OSTAG<sup>3</sup> overcomes the danger of sparse data estimates. Not only does such a model lead to better parsing performance, but it uses adjunction more extensively than its more lightly parameterized alternatives. While different representations make direct

comparison inappropriate, the OSTAG results lie in the same range as previous work with statistical TIG on this task, such as Chiang (2000) (86.00) and Shindo et al. (2011) (85.03).

The OSTAG constraint can be relaxed as described in Section 4.2 to allow any finite number of on-spine adjunctions without sacrificing context-free form. However, the increase to the grammar constant quickly makes parsing with such models an arduous task. To determine the effect of such a relaxation, we allow a single level of on-spine adjunction using the adjunction model of OSTAG<sup>1</sup>, and estimate this model with EM on the training data. We parse sentences of length 40 or less in section 23 and observe that on-spine adjunction is never used in the MPD parses. This suggests that the OSTAG constraint is reasonable, at least for the domain of English news text.

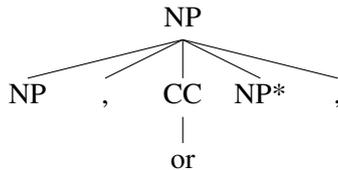
We performed further examination of the MPD using OSTAG for each of the sentences in the test corpus. As an artifact of the English language, the majority have their foot node on the left spine and would also be usable by TIG, and so we discuss the instances of wrapping auxiliary trees in these derivations that are uniquely available to OSTAG. We remove binarization for clarity and denote the foot node with an asterisk.

A frequent use of wrapping adjunction is to coordinate symbols such as quotes, parentheses, and dashes on both sides of a noun phrase. One common wrapping auxiliary tree in our experiments is



This is used frequently in the news text of the Wall Street Journal for reported speech when avoiding a full quotation. This sentence is an example of the way the rule is employed, using what Joshi and Schabes (1997) referred to as ‘‘factoring recursion from linguistic constraints’’ with TAG. Note that replacing the quoted noun phrase and its following prepositional phrase with the noun phrase itself yields a valid sentence, in line with the linguistic theory underlying TAG.

Another frequent wrapping rule, shown below, allows direct coordination between the contents of an appositive with the rest of the sentence.



This is a valuable ability, as it is common to use an appositive to provide context or explanation for a proper noun. As our information on proper nouns will most likely be very sparse, the appositive may be more reliably connected to the rest of the sentence. An example of this from one of the sentences in which this rule appears in the MPD is the phrase “since the market fell 156.83, or 8 %, a week after Black Monday”. The wrapping rule allows us to coordinate the verb “fell” with the pattern “X %” instead of 156.83, which is mapped to an unknown word category.

These rules highlight the linguistic intuitions that back TAG; if their adjunction were undone, the remaining derivation would be a valid sentence that simply lacks the modifying structure of the auxiliary tree. However, the MPD parses reveal that not all useful adjunctions conform to this paradigm, and that left-auxiliary trees that are not used for sister adjunction are susceptible to this behavior. The most common such tree is used to create noun phrases such as

P&G’s share of [the Japanese market]  
 the House’s repeal of [a law]  
 Apple’s family of [Macintosh Computers]  
 Canada’s output of [crude oil]

by adjoining the shared unbracketed syntax onto the NP dominating the bracketed text. If adjunction is taken to model modification, this rule drastically changes the semantics of the unmodified sentence. Furthermore, in some cases removing the adjunction can leave a grammatically incorrect sentence, as in the third example where the noun phrase changes plurality.

While our grammar induction method is a crude (but effective) heuristic, we can still highlight the qualities of the more important auxiliary trees by examining aggregate statistics over the MPD parses, shown in Figure 6. The use of left-auxiliary trees for sister adjunction is a clear trend, as is the predominant use of right-auxiliary trees for the complementary set of “regular” adjunctions, which is to be expected in a right branching language such as English. The statistics also

	All	Wrap	Right	Left
Total	3585 (1374)	41 (26)	1698 (518)	1846 (830)
Sister	2851 (1180)	17 (11)	1109 (400)	1725 (769)
Lex	2244 (990)	28 (19)	894 (299)	1322 (672)
FLex	1028 (558)	7 (2)	835 (472)	186 (84)

Figure 6: Statistics for MPD auxiliary trees using OSTAG<sup>3</sup>. The columns indicate type of auxiliary tree and the rows correspond respectively to the full set found in the MPD, those that perform sister adjunction, those that are lexicalized, and those that are fully lexicalized. Each cell shows the number of tokens followed by the number of types of auxiliary tree that fit its conditions.

reflect the importance of substitution in right-auxiliary trees, as they must capture the wide variety of right branching modifiers of the English language.

## 6 Conclusion

The OSTAG variant of Tree-Adjoining Grammar is a simple weakly context-free formalism that still provides for all types of adjunction and is a bit more concise than TSG (quadratically so). OSTAG can be reversibly transformed into CFG form, allowing the use of a wide range of well studied techniques in statistical parsing.

OSTAG provides an alternative to TIG as a context-free TAG variant that offers wrapping adjunction in exchange for recursive left/right spine adjunction. It would be interesting to apply both OSTAG and TIG to different languages to determine where the constraints of one or the other are more or less appropriate. Another possibility is the combination of OSTAG with TIG, which would strictly expand the abilities of both approaches.

The most important direction of future work for OSTAG is the development of a principled grammar induction model, perhaps using the same techniques that have been successfully applied to TSG and TIG. In order to motivate this and other related research, we release our implementation of EM and CYK parsing for OSTAG<sup>1</sup>. Our system performs the CFG transform described above and optionally employs coarse to fine pruning and relaxed (finite) limits on the number of spine adjunctions. As a TSG is simply a TAG without adjunction rules, our parser can easily be used as a TSG estimator and parser as well.

<sup>1</sup>[bllip.cs.brown.edu/download/bucketparser.tar](http://bllip.cs.brown.edu/download/bucketparser.tar)

## References

- Eugene Charniak, Mark Johnson, Micha Elsner, Joseph L. Austerweil, David Ellis, Isaac Haxton, Catherine Hill, R. Shrivaths, Jeremy Moore, Michael Pozar, and Theresa Vu. 2006. Multilevel coarse-to-fine PCFG parsing. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Eugene Charniak. 1996. Tree-bank grammars. In *Association for the Advancement of Artificial Intelligence*, pages 1031–1036.
- David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. Association for Computational Linguistics.
- Trevor Cohn and Phil Blunsom. 2010. Blocked inference in bayesian tree substitution grammars. pages 225–230. Association for Computational Linguistics.
- Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556. Association for Computational Linguistics.
- Christy Doran, Dania Egedi, Beth Ann Hockey, Bangalore Srinivas, and Martin Zaidel. 1994. XTAG system: a wide coverage grammar for English. pages 922–928. Association for Computational Linguistics.
- J. Goodman. 2003. Efficient parsing of DOP with PCFG-reductions. *Bod et al. 2003*.
- Saul Gorn. 1965. Explicit definitions and linguistic dominoes. In *Systems and Computer Science*, pages 77–115.
- Rebecca Hwa. 1998. An empirical evaluation of probabilistic lexicalized tree insertion grammars. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 557–563. Association for Computational Linguistics.
- Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69–124. Springer.
- Aravind K Joshi. 1985. *Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions?* University of Pennsylvania.
- Aravind K Joshi. 1987. An introduction to tree adjoining grammars. *Mathematics of Language*, pages 87–115.
- Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. pages 423–430. Association for Computational Linguistics.
- K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, pages 35–56.
- Mehryar Mohri and Mark jan Nederhof. 2000. Regular approximation of context-free grammars through transformation. In *Robustness in language and speech technology*.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics.
- Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, (4):479–513.
- Yves Schabes. 1990. *Mathematical and computational aspects of lexicalized grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.
- Hiroyuki Shindo, Akinori Fujino, and Masaaki Nagata. 2011. Insertion operator for bayesian tree substitution grammars. pages 206–211. Association for Computational Linguistics.
- Elif Yamangil and Stuart M. Shieber. 2012. Estimating compact yet rich tree insertion grammars. pages 110–114. Association for Computational Linguistics.

# Fast and Adaptive Online Training of Feature-Rich Translation Models

Spence Green, Sida Wang, Daniel Cer, and Christopher D. Manning

Computer Science Department, Stanford University

{spenceg, sidaw, danielcer, manning}@stanford.edu

## Abstract

We present a fast and scalable online method for tuning statistical machine translation models with large feature sets. The standard tuning algorithm—MERT—only scales to tens of features. Recent discriminative algorithms that accommodate sparse features have produced smaller than expected translation quality gains in large systems. Our method, which is based on stochastic gradient descent with an adaptive learning rate, scales to millions of features and tuning sets with tens of thousands of sentences, while still converging after only a few epochs. Large-scale experiments on Arabic-English and Chinese-English show that our method produces significant translation quality gains by exploiting sparse features. Equally important is our analysis, which suggests techniques for mitigating overfitting and domain mismatch, and applies to other recent discriminative methods for machine translation.

## 1 Introduction

Sparse, overlapping features such as words and  $n$ -gram contexts improve many NLP systems such as parsers and taggers. Adaptation of discriminative learning methods for these types of features to statistical machine translation (MT) systems, which have historically used idiosyncratic learning techniques for a few dense features, has been an active research area for the past half-decade. However, despite some research successes, feature-rich models are rarely used in annual MT evaluations. For example, among all submissions to the WMT and IWSLT 2012 shared tasks, just one participant tuned more than 30 features (Hasler et al., 2012a). Slow uptake of these methods may be due to implementation complexities, or to practical difficulties of configuring them for specific translation tasks (Gimpel and Smith, 2012; Simianer et al., 2012, *inter alia*).

We introduce a new method for training feature-rich MT systems that is effective yet comparatively easy to implement. The algorithm scales to millions of features and large tuning sets. It optimizes a logistic objective identical to that of PRO (Hopkins and May, 2011) with stochastic gradient descent, although other objectives are possible. The learning rate is set adaptively using AdaGrad (Duchi et al., 2011), which is particularly effective for the mixture of dense and sparse features present in MT models. Finally, feature selection is implemented as efficient  $L_1$  regularization in the forward-backward splitting (FOBOS) framework (Duchi and Singer, 2009). Experiments show that our algorithm converges faster than batch alternatives.

To learn good weights for the sparse features, most algorithms—including ours—benefit from more tuning data, and the natural source is the training bitext. However, the bitext presents two problems. First, it has a single reference, sometimes of lower quality than the multiple references in tuning sets from MT competitions. Second, large bitexts often comprise many text genres (Haddow and Koehn, 2012), a virtue for classical dense MT models but a curse for high dimensional models: bitext tuning can lead to a significant domain adaptation problem when evaluating on standard test sets. Our analysis separates and quantifies these two issues.

We conduct large-scale translation quality experiments on Arabic-English and Chinese-English. As baselines we use MERT (Och, 2003), PRO, and the Moses (Koehn et al., 2007) implementation of  $k$ -best MIRA, which Cherry and Foster (2012) recently showed to work as well as online MIRA (Chiang, 2012) for feature-rich models. The first experiment uses standard tuning and test sets from the NIST OpenMT competitions. The second experiment uses tuning and test sets sampled from the large bitexts. The new method yields significant improvements in both experiments. Our code is included in the Phrasal (Cer et al., 2010) toolkit, which is freely available.

## 2 Adaptive Online Algorithms

Machine translation is an unusual machine learning setting because multiple correct translations exist and decoding is comparatively expensive. When we have a large feature set and therefore want to tune on a large data set, batch methods are infeasible. Online methods can converge faster, and in practice they often find *better* solutions (Liang and Klein, 2009; Bottou and Bousquet, 2011, *inter alia*).

Recall that stochastic gradient descent (SGD), a fundamental online method, updates weights  $w$  according to

$$w_t = w_{t-1} - \eta \nabla \ell_t(w_{t-1}) \quad (1)$$

with loss function<sup>1</sup>  $\ell_t(w)$  of the  $t^{\text{th}}$  example, (sub)gradient of the loss with respect to the parameters  $\nabla \ell_t(w_{t-1})$ , and learning rate  $\eta$ .

SGD is sensitive to the learning rate  $\eta$ , which is difficult to set in an MT system that mixes frequent “dense” features (like the language model) with sparse features (e.g., for translation rules). Furthermore,  $\eta$  applies to each coordinate in the gradient, an undesirable property in MT where good sparse features may fire very infrequently. We would instead like to take larger steps for sparse features and smaller steps for dense features.

### 2.1 AdaGrad

AdaGrad is a method for setting an *adaptive learning rate* that comes with good theoretical guarantees. The theoretical improvement over SGD is most significant for high-dimensional, sparse features. AdaGrad makes the following update:

$$w_t = w_{t-1} - \eta \Sigma_t^{1/2} \nabla \ell_t(w_{t-1}) \quad (2)$$

$$\begin{aligned} \Sigma_t^{-1} &= \Sigma_{t-1}^{-1} + \nabla \ell_t(w_{t-1}) \nabla \ell_t(w_{t-1})^\top \\ &= \sum_{i=1}^t \nabla \ell_i(w_{i-1}) \nabla \ell_i(w_{i-1})^\top \end{aligned} \quad (3)$$

A diagonal approximation to  $\Sigma$  can be used for a high-dimensional vector  $w_t$ . In this case, AdaGrad is simple to implement and computationally cheap. Consider a single dimension  $j$ , and let scalars  $v_t = w_{t,j}$ ,  $g_t = \nabla_j \ell_t(w_{t-1})$ ,  $G_t = \sum_{i=1}^t g_i^2$ , then the update rule is

$$v_t = v_{t-1} - \eta G_t^{-1/2} g_t \quad (4)$$

$$G_t = G_{t-1} + g_t^2 \quad (5)$$

Compared to SGD, we just need to store  $G_t = \Sigma_{t,j,j}^{-1}$  for each dimension  $j$ .

<sup>1</sup>We specify the loss function for MT in section 3.1.

### 2.2 Prior Online Algorithms in MT

AdaGrad is related to two previous online learning methods for MT.

**MIRA** Chiang et al. (2008) described an adaption of MIRA (Crammer et al., 2006) to MT. MIRA makes the following update:

$$w_t = \arg \min_w \frac{1}{2\eta} \|w - w_{t-1}\|_2^2 + \ell_t(w) \quad (6)$$

The first term expresses *conservativity*: the weight should change as little as possible based on a single example, ensuring that it is never beneficial to overshoot the minimum.

The relationship to SGD can be seen by linearizing the loss function  $\ell_t(w) \approx \ell_t(w_{t-1}) + (w - w_{t-1})^\top \nabla \ell_t(w_{t-1})$  and taking the derivative of (6). The result is exactly (1).

**AROW** Chiang (2012) adapted AROW (Crammer et al., 2009) to MT. AROW models the current weight as a Gaussian centered at  $w_{t-1}$  with covariance  $\Sigma_{t-1}$ , and does the following update upon seeing training example  $x_t$ :

$$\begin{aligned} w_t, \Sigma_t = \\ \arg \min_{w, \Sigma} \frac{1}{\eta} D_{\text{KL}}(\mathcal{N}(w, \Sigma) || \mathcal{N}(w_{t-1}, \Sigma_{t-1})) \\ + \ell_t(w) + \frac{1}{2\eta} x_t^\top \Sigma x_t \end{aligned} \quad (7)$$

The KL-divergence term expresses a more general, directionally sensitive conservativity. Ignoring the third term, the  $\Sigma$  that minimizes the KL is actually  $\Sigma_{t-1}$ . As a result, the first two terms of (7) generalize MIRA so that we may be more conservative in some directions specified by  $\Sigma$ . To see this, we can write out the KL-divergence between two Gaussians in closed form, and observe that the terms involving  $w$  do not interact with the terms involving  $\Sigma$ :

$$\begin{aligned} w_t = \arg \min_w \frac{1}{2\eta} (w - w_{t-1})^\top \Sigma_{t-1}^{-1} (w - w_{t-1}) \\ + \ell_t(w) \end{aligned} \quad (8)$$

$$\begin{aligned} \Sigma_t = \arg \min_{\Sigma} \frac{1}{2\eta} \log \left( \frac{|\Sigma_{t-1}|}{|\Sigma|} \right) + \frac{1}{2\eta} \text{Tr}(\Sigma_{t-1}^{-1} \Sigma) \\ + \frac{1}{2\eta} x_t^\top \Sigma x_t \end{aligned} \quad (9)$$

The third term in (7), called the confidence term, gives us *adaptivity*, the notion that we should have smaller variance in the direction  $v$  as more data  $x_t$

is seen in direction  $v$ . For example, if  $\Sigma$  is diagonal and  $x_t$  are indicator features, the confidence term then says that the weight for a rarer feature should have more variance and vice-versa. Recall that for generalized linear models  $\nabla \ell_t(w) \propto x_t$ ; if we substitute  $x_t = \alpha_t \nabla \ell_t(w)$  into (9), differentiate and solve, we get:

$$\begin{aligned} \Sigma_t^{-1} &= \Sigma_{t-1}^{-1} + x_t x_t^\top \\ &= \Sigma_0^{-1} + \sum_{i=1}^t \alpha_i^2 \nabla \ell_i(w_{i-1}) \nabla \ell_i(w_{i-1})^\top \end{aligned} \quad (10)$$

The precision  $\Sigma_t^{-1}$  generally grows as more data is seen. Frequently updated features receive an especially high precision, whereas the model maintains large variance for rarely seen features.

If we substitute (10) into (8), linearize the loss  $\ell_t(w)$  as before, and solve, then we have the linearized AROW update

$$w_t = w_{t-1} - \eta \Sigma_t \nabla \ell_t(w_{t-1}) \quad (11)$$

which is also an adaptive update with per-coordinate learning rates specified by  $\Sigma_t$  (as opposed to  $\Sigma_t^{1/2}$  in AdaGrad).

### 2.3 Comparing AdaGrad, MIRA, AROW

Compare (3) to (10) and observe that if we set  $\Sigma_0^{-1} = 0$  and  $\alpha_t = 1$ , then the only difference between the AROW update (11) and the AdaGrad update (2) is a square root. Under a constant gradient, AROW decays the step size more aggressively ( $1/t$ ) compared to AdaGrad ( $1/\sqrt{t}$ ), and it is sensitive to the specification of  $\Sigma_0^{-1}$ .

Informally, SGD can be improved in the conservativity direction using MIRA so the updates do not overshoot. Second, SGD can be improved in the adaptivity direction using AdaGrad where the decaying stepsize is more robust and the adaptive stepsize allows better weight updates to features differing in sparsity and scale. Finally, AROW combines both adaptivity and conservativity. For MT, adaptivity allows us to deal with mixed dense/sparse features effectively without specific normalization.

Why do we choose AdaGrad over AROW? MIRA/AROW requires selecting the loss function  $\ell(w)$  so that  $w_t$  can be solved in closed-form, by a quadratic program (QP), or in some other way that is better than linearizing. This usually means choosing a hinge loss. On the other hand, AdaGrad/linearized AROW only requires that the gradient of the loss function can be computed efficiently.

---

### Algorithm 1 Adaptive online tuning for MT.

---

**Require:** Tuning set  $\{f_i, e_i^{1:k}\}_{i=1:M}$

- 1: Set  $w_0 = 0$
- 2: Set  $t = 1$
- 3: **repeat**
- 4:   **for**  $i$  in  $1 \dots M$  in random order **do**
- 5:     Decode  $n$ -best list  $N_i$  for  $f_i$
- 6:     Sample pairs  $\{d_{j,+}, d_{j,-}\}_{j=1:s}$  from  $N_i$
- 7:     Compute  $\mathcal{D}_i = \{\phi(d_{j,+}) - \phi(d_{j,-})\}_{j=1:s}$
- 8:     Set  $g_t = \nabla \ell(\mathcal{D}_i; w_{t-1})$
- 9:     Set  $\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + g_t g_t^\top$  ▷ Eq. (3)
- 10:     Update  $w_t = w_{t-1} - \eta \Sigma_t^{1/2} g_t$  ▷ Eq. (2)
- 11:     Regularize  $w_t$  ▷ Eq. (15)
- 12:     Set  $t = t + 1$
- 13:   **end for**
- 14: **until** convergence

---

Linearized AROW, however, is less robust than AdaGrad empirically<sup>2</sup> and lacks known theoretical guarantees. Finally, by using AdaGrad, we separate adaptivity from conservativity. Our experiments suggest that adaptivity is actually more important.

## 3 Adaptive Online MT

Algorithm 1 shows the full algorithm introduced in this paper. AdaGrad (lines 9–10) is a crucial piece, but the loss function, regularization technique, and parallelization strategy described in this section are equally important in the MT setting.

### 3.1 Pairwise Logistic Loss Function

Algorithm 1 lines 5–8 describe the gradient computation. We cast MT tuning as *pairwise ranking* (Herbrich et al., 1999, *inter alia*), which Hopkins and May (2011) applied to MT. The pairwise approach results in simple, convex loss functions suitable for online learning. The idea is that for any two derivations, the ranking predicted by the model should be consistent with the ranking predicted by a gold sentence-level metric  $G$  like BLEU+1 (Lin and Och, 2004).

Consider a single source sentence  $f$  with associated references  $e^{1:k}$ . Let  $d$  be a derivation in an  $n$ -best list of  $f$  that has the target  $e = e(d)$  and the feature map  $\phi(d)$ . Let  $M(d) = w \cdot \phi(d)$  be the model score. For any derivation  $d_+$  that is better than  $d_-$  under  $G$ , we desire pairwise agreement such that

$$\begin{aligned} G(e(d_+), e^{1:k}) &> G(e(d_-), e^{1:k}) \\ &\iff M(d_+) > M(d_-) \end{aligned}$$

---

<sup>2</sup>According to experiments not reported in this paper.

Ensuring pairwise agreement is the same as ensuring  $w \cdot [\phi(d_+) - \phi(d_-)] > 0$ .

For learning, we need to select derivation pairs  $(d_+, d_-)$  to compute difference vectors  $x_+ = \phi(d_+) - \phi(d_-)$ . Then we have a 1-class separation problem trying to ensure  $w \cdot x_+ > 0$ . The derivation pairs are sampled with the algorithm of Hopkins and May (2011).

We compute difference vectors  $\mathcal{D}_t = \{x_+^{1:s}\}$  (Algorithm 1 line 7) from  $s$  pairs  $(d_+, d_-)$  for source sentence  $f_t$ . We use the familiar logistic loss:

$$\ell_t(w) = \ell(\mathcal{D}_t, w) = - \sum_{x_+ \in \mathcal{D}_t} \log \frac{1}{1 + e^{-w \cdot x_+}} \quad (12)$$

Choosing the hinge loss instead of the logistic loss results in the 1-class SVM problem. The 1-class separation problem is equivalent to the binary classification problem with  $x_+ = \phi(d_+) - \phi(d_-)$  as positive data and  $x_- = -x_+$  as negative data, which may be plugged into an existing logistic regression solver.

We find that Algorithm 1 works best with mini-batches instead of single examples. In line 4 we simply partition the tuning set so that  $i$  becomes a mini-batch of examples.

### 3.2 Updating and Regularization

Algorithm 1 lines 9–11 compute the adaptive learning rate, update the weights, and apply regularization. Section 2.1 explained the AdaGrad learning rate computation. To update and regularize the weights we apply the Forward-Backward Splitting (FOBOS) (Duchi and Singer, 2009) framework, which separates the two operations. The two-step FOBOS update is

$$w_{t-\frac{1}{2}} = w_{t-1} - \eta_{t-1} \nabla \ell_{t-1}(w_{t-1}) \quad (13)$$

$$w_t = \arg \min_w \frac{1}{2} \|w - w_{t-\frac{1}{2}}\|_2^2 + \eta_{t-1} r(w) \quad (14)$$

where (13) is just an unregularized gradient descent step and (14) balances the regularization term  $r(w)$  with staying close to the gradient step.

Equation (14) permits efficient  $L_1$  regularization, which is well-suited for selecting good features from exponentially many irrelevant features (Ng, 2004). It is well-known that feature selection is very important for feature-rich MT. For example, simple indicator features like lexicalized re-ordering classes are potentially useful yet bloat the feature set and, in the worst case, can negatively impact

---

### Algorithm 2 “Stale gradient” parallelization method for Algorithm 1.

---

**Require:** Tuning set  $\{f_i, e_i^{1:k}\}_{i=1:M}$

- 1: Initialize threadpool  $p_1, \dots, p_j$
- 2: Set  $t = 1$
- 3: **repeat**
- 4:   **for**  $i$  in  $1 \dots M$  in random order **do**
- 5:     Wait until any thread  $p$  is idle
- 6:     Send  $(f_i, e_i^{1:k}, t)$  to  $p$   $\triangleright$  Alg. 1 lines 5–8
- 7:     **while**  $\exists p'$  done with gradient  $g_{t'}$  **do**  $\triangleright t' \leq t$
- 8:       Update  $w_t = w_{t-1} - \eta g_{t'}$   $\triangleright$  Alg. 1 lines 9–11
- 9:       Set  $t = t + 1$
- 10:     **end while**
- 11:   **end for**
- 12: **until** convergence

---

search. Some of the features generalize, but many do not. This was well understood in previous work, so heuristic filtering was usually applied (Chiang et al., 2009, *inter alia*). In contrast, we need only select an appropriate regularization strength  $\lambda$ .

Specifically, when  $r(w) = \lambda \|w\|_1$ , the closed-form solution to (14) is

$$w_t = \text{sign}(w_{t-\frac{1}{2}}) \left[ |w_{t-\frac{1}{2}}| - \eta_{t-1} \lambda \right]_+ \quad (15)$$

where  $[x]_+ = \max(x, 0)$  is the clipping function that in this case sets a weight to 0 when it falls below the threshold  $\eta_{t-1} \lambda$ . It is straightforward to adapt this to AdaGrad with diagonal  $\Sigma$  by setting each dimension of  $\eta_{t-1,j} = \eta \Sigma_{t,jj}^{\frac{1}{2}}$  and by taking element-wise products.

We find that  $\nabla \ell_{t-1}(w_{t-1})$  only involves several hundred active features for the current example (or mini-batch). However, naively following the FOBOS framework requires updating millions of weights. But a practical benefit of FOBOS is that we can do lazy updates on just the active dimensions without any approximations.

### 3.3 Parallelization

Algorithm 1 is inherently sequential like standard online learning. This is undesirable in MT where decoding is costly. We therefore parallelize the algorithm with the “stale gradient” method of Langford et al. (2009) (Algorithm 2). A fixed threadpool of workers computes gradients in parallel and sends them to a master thread, which updates a central weight vector. Crucially, the weight updates need not be applied in order, so synchronization is unnecessary; the workers only idle at the end of an epoch. The consequence is that the update in line 8 of Algorithm 2 is with respect to gradient  $g_{t'}$  with  $t' \leq t$ . Langford et al. (2009) gave convergence results for

stale updating, but the bounds do not apply to our setting since we use  $L_1$  regularization. Nevertheless, Gimpel et al. (2010) applied this framework to other non-convex objectives and obtained good empirical results.

Our asynchronous, stochastic method has practical appeal for MT. During a tuning run, the online method decodes the tuning set under many more weight vectors than a MERT-style batch method. This characteristic may result in broader exploration of the search space, and make the learner more robust to local optima (Liang and Klein, 2009; Bottou and Bousquet, 2011, *inter alia*). The adaptive algorithm identifies appropriate learning rates for the mixture of dense and sparse features. Finally, large data structures such as the language model (LM) and phrase table exist in shared memory, obviating the need for remote queries.

## 4 Experiments

We built Arabic-English and Chinese-English MT systems with Phrasal (Cer et al., 2010), a phrase-based system based on alignment templates (Och and Ney, 2004). The corpora<sup>3</sup> in our experiments (Table 1) derive from several LDC sources from 2012 and earlier. We de-duplicated each bitext according to exact string match, and ensured that no overlap existed with the test sets. We produced alignments with the Berkeley aligner (Liang et al., 2006b) with standard settings and symmetrized via the grow-diag heuristic.

For each language we used SRILM (Stolcke, 2002) to estimate 5-gram LMs with modified Kneser-Ney smoothing. We included the monolingual English data and the respective target bitexts.

### 4.1 Feature Templates

The baseline “dense” model contains 19 features: the nine Moses baseline features, the hierarchical lexicalized re-ordering model of Galley and Manning (2008), the (log) count of each rule, and an indicator for unique rules.

To the dense features we add three high dimensional “sparse” feature sets. **Discrimina-**

<sup>3</sup>We tokenized the English with packages from the Stanford Parser (Klein and Manning, 2003) according to the Penn Treebank standard (Marcus et al., 1993), the Arabic with the Stanford Arabic segmenter (Green and DeNero, 2012) according to the Penn Arabic Treebank standard (Maamouri et al., 2008), and the Chinese with the Stanford Chinese segmenter (Chang et al., 2008) according to the Penn Chinese Treebank standard (Xue et al., 2005).

	Bilingual		Monolingual
	Sentences	Tokens	Tokens
Ar-En	6.6M	375M	990M
Zh-En	9.3M	538M	

Table 1: Bilingual and monolingual corpora used in these experiments. The monolingual English data comes from the AFP and Xinhua sections of English Gigaword 4 (LDC2009T13).

**tive phrase table (PT)**: indicators for each rule in the phrase table. **Alignments (AL)**: indicators for phrase-internal alignments and deleted (unaligned) source words. **Discriminative re-ordering (LO)**: indicators for eight lexicalized re-ordering classes, including the six standard monotone/swap/discontinuous classes plus the two simpler Moses monotone/non-monotone classes.

### 4.2 Tuning Algorithms

The primary baseline is the dense feature set tuned with MERT (Och, 2003). The Phrasal implementation uses the line search algorithm of Cer et al. (2008), uniform initialization, and 20 random starting points.<sup>4</sup> We tuned according to BLEU-4 (Papineni et al., 2002).

We built high dimensional baselines with two different algorithms. First, we tuned with batch PRO using the default settings in Phrasal ( $L_2$  regularization with  $\sigma=0.1$ ). Second, we ran the  $k$ -best batch MIRA (kb-MIRA) (Cherry and Foster, 2012) implementation in Moses. We did implement an online version of MIRA, and in small-scale experiments found that the batch variant worked just as well. Cherry and Foster (2012) reported the same result, and their implementation is available in Moses. We ran their code with standard settings.

Moses<sup>5</sup> also contains the discriminative phrase table implementation of (Hasler et al., 2012b), which is identical to our implementation using Phrasal. Moses and Phrasal accept the same phrase table and LM formats, so we kept those data structures in common. The two decoders also use the same multi-stack beam search (Och and Ney, 2004).

For our method, we used uniform initialization, 16 threads, and a mini-batch size of 20. We found that  $\eta=0.02$  and  $\lambda=0.1$  worked well on development sets for both languages. To compute the gradients

<sup>4</sup>Other system settings for all experiments: distortion limit of 5, a maximum phrase length of 7, and an  $n$ -best size of 200.  
<sup>5</sup>v1.0 (28 January 2013)

Model	#features	Algorithm	Tuning Set	MT02	MT03	MT04	MT09	
Dense	19	MERT	MT06	45.08	51.32	52.26	51.42	48.44
Dense	19	This paper	MT06	44.19	51.42	52.52	50.16	48.13
+PT	151k	kb-MIRA	MT06	42.08	47.25	48.98	47.08	45.64
+PT	23k	PRO	MT06	44.31	51.06	52.18	50.23	47.52
+PT	50k	This paper	MT06	50.61	<b>51.71</b>	52.89	50.42	<b>48.74</b>
+PT+AL+LO	109k	PRO	MT06	44.87	51.25	52.43	50.05	47.76
+PT+AL+LO	242k	This paper	MT06	57.84	52.45	53.18	51.38	49.37
Dense	19	MERT	MT05/6/8	49.63	51.60	52.29	51.73	48.68
+PT+AL+LO	390k	This paper	MT05/6/8	58.20	<b>53.61</b>	<b>54.99</b>	<b>52.79</b>	<b>49.94</b>
(Chiang, 2012)*	10-20k	MIRA	MT04/6	–	–	–	–	45.90
(Chiang, 2012)*	10-20k	AROW	MT04/6	–	–	–	–	47.60
				<i>#sentences</i>	728	663	1,075	1,313

Table 2: Ar-En results [BLEU-4 % uncased] for the NIST tuning experiment. The tuning and test sets each have four references. MT06 has 1,717 sentences, while the concatenated MT05/6/8 set has 4,213 sentences. **Bold** indicates statistical significance relative to the *best* baseline in each block at  $p < 0.001$ ; *bold-italic* at  $p < 0.05$ . We assessed significance with the permutation test of Riezler and Maxwell (2005). (\*) Chiang (2012) used a similar-sized bitext, but two LMs trained on twice as much monolingual data.

Model	#features	Algorithm	Tuning Set	MT02	MT03	MT04	
Dense	19	MERT	MT06	33.90	35.72	33.71	34.26
Dense	19	This paper	MT06	32.60	36.23	<b>35.14</b>	<b>34.78</b>
+PT	105k	kb-MIRA	MT06	29.46	30.67	28.96	30.05
+PT	26k	PRO	MT06	33.70	36.87	34.62	34.80
+PT	66k	This paper	MT06	33.90	36.09	34.86	34.73
+PT+AL+LO	148k	PRO	MT06	34.81	36.31	33.81	34.41
+PT+AL+LO	344k	This paper	MT06	38.99	36.40	35.07	34.84
Dense	19	MERT	MT05/6/8	32.36	35.69	33.83	34.33
+PT+AL+LO	487k	This paper	MT05/6/8	37.64	<b>37.81</b>	<b>36.26</b>	<b>36.15</b>
				<i>#sentences</i>	878	919	1,597

Table 3: Zh-En results [BLEU-4 % uncased] for the NIST tuning experiment. MT05/6/8 has 4,103 sentences. OpenMT 2009 did not include Zh-En, hence the asymmetry with Table 2.

we sampled 15 derivation pairs for each tuning example and scored them with BLEU+1.

### 4.3 NIST OpenMT Experiment

The first experiment evaluates our algorithm when tuning and testing on standard test sets, each with four references. When we add features, our algorithm tends to overfit to a standard-sized tuning set like MT06. We thus concatenated MT05, MT06, and MT08 to create a larger tuning set.

Table 2 shows the Ar-En results. Our algorithm is competitive with MERT in the low dimensional “dense” setting, and compares favorably to PRO

with the PT feature set. PRO does not benefit from additional features, whereas our algorithm improves with both additional features and data. The underperformance of kb-MIRA may result from a difference between Moses and Phrasal: Moses MERT achieves only 45.62 on MT09. Moses PRO with the PT feature set is slightly worse, e.g., 44.52 on MT09. Nevertheless, kb-MIRA does not improve significantly over MERT, and also selects an unnecessarily large model.

The full feature set PT+AL+LO does help. With the PT feature set alone, our algorithm tuned on MT05/6/8 scores well below the best model, e.g.

Model	#features	Algorithm	Tuning Set	#refs	bitext5k-test	MT04	
Dense	19	MERT	MT06	45.08	4	39.28	51.42
+PT	72k	This paper	MT05/6/8	51.29	4	39.50	50.60
+PT	79k	This paper	bitext5k	44.79	1	43.85	45.73
+PT+AL+LO	647k	This paper	bitext15k	45.68	1	<b>43.93</b>	45.24

Table 4: Ar-En results [BLEU-4 % uncased] for the bitext tuning experiment. Statistical significance is relative to the Dense baseline. We include MT04 for comparison to the NIST genre.

Model	#features	Algorithm	Tuning Set	#refs	bitext5k-test	MT04	
Dense	19	MERT	MT06	33.90	4	33.44	34.26
+PT	97k	This paper	MT05/6/8	34.45	4	35.08	35.19
+PT	67k	This paper	bitext5k	36.26	1	36.01	33.76
+PT+AL+LO	536k	This paper	bitext15k	37.57	1	<b>36.30</b>	34.05

Table 5: Zh-En results [BLEU-4 % uncased] for the bitext tuning experiment.

48.56 BLEU on MT09. For Ar-En, our algorithm thus has the desirable property of benefiting from more and better features, and more data.

Table 3 shows Zh-En results. Somewhat surprisingly our algorithm improves over MERT in the dense setting. When we add the discriminative phrase table, our algorithm improves over kb-MIRA, and over batch PRO on two evaluation sets. With all features and the MT05/6/8 tuning set, we improve significantly over all other models. PRO learns a smaller model with the PT+AL+LO feature set which is surprising given that it applies  $L_2$  regularization (AdaGrad uses  $L_1$ ). We speculate that this may be a consequence of stochastic learning. Our algorithm decodes each example with a new weight vector, thus exploring more of the search space for the same tuning set.

#### 4.4 Bitext Tuning Experiment

Tables 2 and 3 show that adding tuning examples improves translation quality. Nevertheless, even the larger tuning set is small relative to the bitext from which rules were extracted. He and Deng (2012) and Simianer et al. (2012) showed significant translation quality gains by tuning on the bitext. However, their bitexts matched the genre of their test sets. Our bitexts, like those of most large-scale systems, do not. Domain mismatch matters for the dense feature set (Haddow and Koehn, 2012). We show that it also matters for feature-rich MT.

Before aligning each bitext, we randomly sampled and sequestered 5k and 15k sentence tuning sets, and a 5k test set. We prevented overlap be-

$\mathcal{D}_A$	$\mathcal{D}_B$	$ A $	$ B $	$ A \cap B $
MT04	MT06	70k	72k	5.9k
MT04	MT568	70k	96k	7.6k
MT04	bitext5k	70k	67k	4.4k
MT04	bitext15k	70k	310k	10.5k
5ktest	bitext5k	82k	67k	5.6k
5ktest	bitext15k	82k	310k	14k

Table 6: Number of overlapping phrase table (+PT) features on various Zh-En dataset pairs.

tween the tuning sets and the test set. We then tuned a dense model with MERT on MT06, and feature-rich models on both MT05/6/8 and the bitext tuning set. Table 4 shows the Ar-En results. When tuned on bitext5k the translation quality gains are significant for bitext5k-test relative to tuning on MT05/6/8, which has multiple references. However, the bitext5k models do not generalize as well to the NIST evaluation sets as represented by the MT04 result. Table 5 shows similar trends for Zh-En.

## 5 Analysis

### 5.1 Feature Overlap Analysis

How many sparse features appear in both the tuning and test sets? In Table 6,  $A$  is the set of phrase table features that received a non-zero weight when tuned on dataset  $\mathcal{D}_A$  (same for  $B$ ). Column  $\mathcal{D}_A$  lists several Zh-En test sets used and column  $\mathcal{D}_B$  lists tuning sets. Our experiments showed that tuning on MT06 generalizes better to MT04 than tuning

on bitext5k, whereas tuning on bitext5k generalizes better to bitext5k-test than tuning on MT06. These trends are consistent with the level of feature overlap. Phrase table features in  $A \cap B$  are overwhelmingly short, simple, and correct phrases, suggesting  $L_1$  regularization is effective for feature selection. It is also important to balance the number of features with how well weights can be learned for those features, as tuning on bitext15k produced higher coverage for MT04 but worse generalization than tuning on MT06.

## 5.2 Domain Adaptation Analysis

To understand the domain adaptation issue we compared the non-zero weights in the discriminative phrase table (PT) for Ar-En models tuned on bitext5k and MT05/6/8. Table 7 illustrates a statistical idiosyncrasy in the data for the American and British spellings of program/programme. The mass is concentrated along the diagonal, probably because MT05/6/8 was prepared by NIST, an American agency, while the bitext was collected from many sources including Agence France Presse.

Of course, this discrepancy is consequential for both dense and feature-rich models. However, we observe that the feature-rich models fit the tuning data more closely. For example, the MT05/6/8 model learns rules like يتضمن برنامج  $\rightarrow$  *program includes*, برنامج  $\rightarrow$  *program of*, and نافذة البرنامج  $\rightarrow$  *program window*. Crucially, it does not learn the basic rule برنامج  $\rightarrow$  *program*.

In contrast, the bitext5k model contains basic rules such برنامج  $\rightarrow$  *programme*, هذا البرنامج  $\rightarrow$  *this programme*, and ذلك البرنامج  $\rightarrow$  *that programme*. It also contains more elaborate rules such as كانت نفقات البرنامج  $\rightarrow$  *programme expenses were* and برامج الرحلات الفضائية المأهولة  $\rightarrow$  *manned space flight programmes*. We observed similar trends for ‘defense/defence’, ‘analyze/analyse’, etc. This particular genre problem could be addressed with language-specific pre-processing, but our system solves it in a data-driven manner.

## 5.3 Re-ordering Analysis

We also analyzed re-ordering differences. Arabic matrix clauses tend to be verb-initial, meaning that the subject and verb must be swapped when translating to English. To assess re-ordering differences—if any—between the dense and feature-rich models, we selected all MT09 segments that began with one

	# bitext5k	# MT05/6/8
<i>programme</i>	185	0
<i>program</i>	19	449
PT rules w/ <i>programme</i>	353	79
PT rules w/ <i>program</i>	9	31

Table 7: Top: comparison of token counts in two Ar-En tuning sets for *programme* and *program*. Bottom: rule counts in the discriminative phrase table (PT) for models tuned on the two tuning sets. Both spellings correspond to the Arabic برنامج.

of seven common verbs: قال *qaal* ‘said’, صرح *SrH* ‘declared’, أشار *ashaar* ‘indicated’, كان *kaan* ‘was’, أعلِن *a<sup>c</sup>ln* ‘announced’, أضاف *aDaaf* ‘added’, ذكر *dhkr* ‘commented’. We compared the output of the MERT Dense model to our method with the full feature set, both tuned on MT06. Of the 208 source segments, 32 of the translation pairs contained different word order in the matrix clause. Our feature-rich model was correct 18 times (56.3%), Dense was correct 4 times (12.5%), and neither method was correct 10 times (31.3%).

- (1) ref: lebanese prime minister , fuad siniora , announced
  - a. and lebanese prime minister fuad siniora that
  - b. the lebanese prime minister fouad siniora announced
- (2) ref: the newspaper and television reported
  - a. she said the newspaper and television
  - b. television and newspaper said

In (1) the dense model (1a) drops the verb while the feature-rich model correctly re-orders and inserts it after the subject (1b). The coordinated subject in (2) becomes an embedded subject in the dense output (2a). The feature-rich model (2b) performs the correct re-ordering.

## 5.4 Runtime Comparison

Table 8 compares our method to standard implementations of the other algorithms. MERT parallelizes easily but runtime increases quadratically with  $n$ -best list size. PRO runs (single-threaded) L-BFGS to convergence on every epoch, a potentially slow procedure for the larger feature set. Moreover, both

		epochs	min.
MERT	Dense	22	180
PRO	+PT	25	35
kb-MIRA*	+PT	26	25
This paper	+PT	10	10
PRO	+PT+AL+LO	13	150
This paper	+PT+AL+LO	5	15

Table 8: Epochs to convergence (“epochs”) and approximate runtime per epoch in minutes (“min.”) for selected Zh-En experiments tuned on MT06. All runs executed on the same dedicated system with the same number of threads. (\*) Moses and kb-MIRA are written in C++, while all other rows refer to Java implementations in Phrasal.

the Phrasal and Moses PRO implementations use  $L_2$  regularization, which regularizes every weight on every update. kb-MIRA makes multiple passes through the  $n$ -best lists during each epoch. The Moses implementation parallelizes decoding but weight updating is sequential.

The core of our method is an inner product between the adaptive learning rate vector and the gradient. This is easy to implement and is very fast even for large feature sets. Since we applied lazy regularization, this inner product usually involves hundred-dimensional vectors. Finally, our method does not need to accumulate  $n$ -best lists, a practice that slows down the other algorithms.

## 6 Related Work

Our work relates most closely to that of Hasler et al. (2012b), who tuned models containing both sparse and dense features with Moses. A discriminative phrase table helped them improve slightly over a dense, online MIRA baseline, but their best results required initialization with MERT-tuned weights and re-tuning a single, shared weight for the discriminative phrase table with MERT. In contrast, our algorithm learned good high dimensional models from a uniform starting point.

Chiang (2012) adapted AROW to MT and extended previous work on online MIRA (Chiang et al., 2008; Watanabe et al., 2007). It was not clear if his improvements came from the novel Hope/Fear search, the conservativity gain from MIRA/AROW by solving the QP exactly, adaptivity, or sophisticated parallelization. In contrast, we show that

AdaGrad, which ignores conservativity and only capturing adaptivity, is sufficient.

Simianer et al. (2012) investigated SGD with a pairwise perceptron objective. Their best algorithm used *iterative parameter mixing* (McDonald et al., 2010), which we found to be slower than the stale gradient method in section 3.3. They regularized once at the end of each epoch, whereas we regularized each weight update. An empirical comparison of these two strategies would be an interesting future contribution.

Watanabe (2012) investigated SGD and even randomly selected pairwise samples as we did. He considered both softmax and hinge losses, observing better results with the latter, which solves a QP. Their parallelization strategy required a line search at the end of each epoch.

Many other discriminative techniques have been proposed based on: ramp loss (Gimpel, 2012); hinge loss (Cherry and Foster, 2012; Haddow et al., 2011; Arun and Koehn, 2007); maximum entropy (Xiang and Ittycheriah, 2011; Ittycheriah and Roukos, 2007; Och and Ney, 2002); perceptron (Liang et al., 2006a); and structured SVM (Tillmann and Zhang, 2006). These works use radically different experimental setups, and to our knowledge only (Cherry and Foster, 2012) and this work compare to at least two high dimensional baselines. Broader comparisons, though time-intensive, could help differentiate these methods.

## 7 Conclusion and Outlook

We introduced a new online method for tuning feature-rich translation models. The method is faster per epoch than MERT, scales to millions of features, and converges quickly. We used efficient  $L_1$  regularization for feature selection, obviating the need for the feature scaling and heuristic filtering common in prior work. Those comfortable with implementing vanilla SGD should find our method easy to implement. Even basic discriminative features were effective, so we believe that our work enables fresh approaches to more sophisticated MT feature engineering.

**Acknowledgments** We thank John DeNero for helpful comments on an earlier draft. The first author is supported by a National Science Foundation Graduate Research Fellowship. We also acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Broad Operational Language Translation (BOLT) program through IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the DARPA or the US government.

## References

- A. Arun and P. Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. In *MT Summit XI*.
- L. Bottou and O. Bousquet. 2011. The tradeoffs of large scale learning. In *Optimization for Machine Learning*, pages 351–368. MIT Press.
- D. Cer, D. Jurafsky, and C. D. Manning. 2008. Regularization and search for minimum error rate training. In *WMT*.
- D. Cer, M. Galley, D. Jurafsky, and C. D. Manning. 2010. Phrasal: A statistical machine translation toolkit for exploring new model features. In *HLT-NAACL, Demonstration Session*.
- P-C. Chang, M. Galley, and C. D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *WMT*.
- C. Cherry and G. Foster. 2012. Batch tuning strategies for statistical machine translation. In *HLT-NAACL*.
- D. Chiang, Y. Marton, and P. Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *EMNLP*.
- D. Chiang, K. Knight, and W. Wang. 2009. 11,001 new features for statistical machine translation. In *HLT-NAACL*.
- D. Chiang. 2012. Hope and fear for discriminative training of statistical translation models. *JMLR*, 13:1159–1187.
- K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive-aggressive algorithms. *JMLR*, 7:551–585.
- K. Crammer, A. Kulesza, and M. Dredze. 2009. Adaptive regularization of weight vectors. In *NIPS*.
- J. Duchi and Y. Singer. 2009. Efficient online and batch learning using forward backward splitting. *JMLR*, 10:2899–2934.
- J. Duchi, E. Hazan, and Y. Singer. 2011. Adaptive sub-gradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159.
- M. Galley and C. D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *EMNLP*.
- K. Gimpel and N. A. Smith. 2012. Structured ramp loss minimization for machine translation. In *HLT-NAACL*.
- K. Gimpel, D. Das, and N. A. Smith. 2010. Distributed asynchronous online learning for natural language processing. In *CoNLL*.
- K. Gimpel. 2012. *Discriminative Feature-Rich Modeling for Syntax-Based Machine Translation*. Ph.D. thesis, Language Technologies Institute, Carnegie Mellon University.
- S. Green and J. DeNero. 2012. A class-based agreement model for generating accurately inflected translations. In *ACL*.
- B. Haddow and P. Koehn. 2012. Analysing the effect of out-of-domain data on SMT systems. In *WMT*.
- B. Haddow, A. Arun, and P. Koehn. 2011. SampleRank training for phrase-based machine translation. In *WMT*.
- E. Hasler, P. Bell, A. Ghoshal, B. Haddow, P. Koehn, F. McInnes, et al. 2012a. The UEDIN systems for the IWSLT 2012 evaluation. In *IWSLT*.
- E. Hasler, B. Haddow, and P. Koehn. 2012b. Sparse lexicalised features and topic adaptation for SMT. In *IWSLT*.
- X. He and L. Deng. 2012. Maximum expected BLEU training of phrase and lexicon translation models. In *ACL*.
- R. Herbrich, T. Graepel, and K. Obermayer. 1999. Support vector learning for ordinal regression. In *ICANN*.
- M. Hopkins and J. May. 2011. Tuning as ranking. In *EMNLP*.
- A. Ittycheriah and S. Roukos. 2007. Direct translation model 2. In *HLT-NAACL*.
- D. Klein and C. D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*.
- P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL, Demonstration Session*.
- J. Langford, A. J. Smola, and M. Zinkevich. 2009. Slow learners are fast. In *NIPS*.
- P. Liang and D. Klein. 2009. Online EM for unsupervised models. In *HLT-NAACL*.
- P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar. 2006a. An end-to-end discriminative approach to machine translation. In *ACL*.
- P. Liang, B. Taskar, and D. Klein. 2006b. Alignment by agreement. In *NAACL*.
- C.-Y. Lin and F. J. Och. 2004. ORANGE: a method for evaluating automatic evaluation metrics for machine translation. In *COLING*.
- M. Maamouri, A. Bies, and S. Kulick. 2008. Enhancing the Arabic Treebank: A collaborative effort toward new annotation guidelines. In *LREC*.

- M. Marcus, M. A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.
- R. McDonald, K. Hall, and G. Mann. 2010. Distributed training strategies for the structured perceptron. In *NAACL-HLT*.
- A. Y. Ng. 2004. Feature selection,  $L_1$  vs.  $L_2$  regularization, and rotational invariance. In *ICML*.
- F. J. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *ACL*.
- F. J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.
- F. J. Och. 2003. Minimum error rate training for statistical machine translation. In *ACL*.
- K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL*.
- S. Riezler and J. T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing in MT. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization (MTSE)*.
- P. Simianer, S. Riezler, and C. Dyer. 2012. Joint feature selection in distributed stochastic learning for large-scale discriminative training in SMT. In *ACL*.
- A. Stolcke. 2002. SRILM—an extensible language modeling toolkit. In *ICSLP*.
- C. Tillmann and T. Zhang. 2006. A discriminative global training algorithm for statistical MT. In *ACL-COLING*.
- T. Watanabe, J. Suzuki, H. Tsukada, and H. Isozaki. 2007. Online large-margin training for statistical machine translation. In *EMNLP-CoNLL*.
- T. Watanabe. 2012. Optimized online rank learning for machine translation. In *HLT-NAACL*. Association for Computational Linguistics.
- B. Xiang and A. Ittycheriah. 2011. Discriminative feature-tied mixture modeling for statistical machine translation. In *ACL-HLT*.
- N. Xue, F. Xia, F. Chiou, and M. Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

# Advancements in Reordering Models for Statistical Machine Translation

Minwei Feng and Jan-Thorsten Peter and Hermann Ney

Human Language Technology and Pattern Recognition

Computer Science Department, RWTH Aachen University, Aachen, Germany

<surname>@cs.rwth-aachen.de

## Abstract

In this paper, we propose a novel reordering model based on sequence labeling techniques. Our model converts the reordering problem into a sequence labeling problem, i.e. a tagging task. Results on five Chinese-English NIST tasks show that our model improves the baseline system by 1.32 BLEU and 1.53 TER on average. Results of comparative study with other seven widely used reordering models will also be reported.

## 1 Introduction

The systematic word order difference between two languages poses a challenge for current statistical machine translation (SMT) systems. The system has to decide in which order to translate the given source words. This problem is known as the reordering problem. As shown in (Knight, 1999), if arbitrary reordering is allowed, the search problem is NP-hard.

Many ideas have been proposed to address the reordering problem. Within the phrase-based SMT framework there are mainly three stages where improved reordering could be integrated:

**In the preprocessing:** the source sentence is reordered by heuristics, so that the word order of source and target sentences is similar. (Wang et al., 2007) use manually designed rules to reorder parse trees of the source sentences. Based on shallow syntax, (Zhang et al., 2007) use rules to reorder the source sentences on the chunk level and provide a source-reordering lattice instead of a single reordered source sentence as input to the SMT system. Designing rules to reorder the source sentence is conceptually clear and usually easy to implement. In this way, syntax information can be incorporated into phrase-based SMT systems. However, one disadvantage is that the reliability of the rules is often language pair dependent.

**In the decoder:** we can add *constraints* or *models* into the decoder to reward good reordering options or penalize bad ones. For reordering constraints, early work includes ITG constraints (Wu, 1995) and IBM constraints (Berger et al., 1996). (Zens and Ney, 2003) did comparative study over different reordering constraints. This paper focuses on reordering models. For reordering models, we can further roughly divide the existing methods into three genres:

- *The reordering is a classification problem.* The classifier will make decision on next phrase's relative position with current phrase. The classifier can be trained with maximum likelihood like Moses lexicalized reordering (Koehn et al., 2007) and hierarchical lexicalized reordering model (Galley and Manning, 2008) or be trained under maximum entropy framework (Zens and Ney, 2006).
- *The reordering is a decoding order problem.* (Mariño et al., 2006) present a translation model that constitutes a language model of a sort of bilanguage composed of bilingual units. From the reordering point of view, the idea is that the correct reordering is a suitable order of translation units. (Feng et al., 2010) present a simpler version of (Mariño et al., 2006)'s model which utilize only source words to model the decoding order.
- *The reordering can be solved by outside heuristics.* We can put human knowledge into the decoder. For example, the simple jump model using linear distance tells the decoder that usually the long range reordering should be avoided. (Cherry, 2008) uses information from dependency trees to make the decoding process keep syntactic cohesion. (Feng et al., 2012) present a method that utilizes predicate-argument structures from semantic role labeling results as soft constraints.

**In the reranking framework:** in principle, all

the models in previous category can be used in the reranking framework, because in the reranking we have all the information (source and target words/phrases, alignment) about the translation process. (Och et al., 2004) describe the use of syntactic features in the rescoring step. However, they report the syntactic features contribute very small gains. One disadvantage of carrying out reordering in reranking is the representativeness of the N-best list is often a question mark.

In this paper, we propose a novel tagging style reordering model which is under the category “*The reordering is a decoding order problem*”. Our model converts the decoding order problem into a sequence labeling problem, i.e. a tagging task. The remainder of this paper is organized as follows: Section 2 introduces the basement of this research: the principle of statistical machine translation. Section 3 describes the proposed model. Section 4 briefly describes several reordering models with which we compare our method. Section 5 provides the experimental configuration and results. Conclusion will be given in Section 6.

## 2 Translation System Overview

In statistical machine translation, we are given a source language sentence  $f_1^J = f_1 \dots f_j \dots f_J$ . The objective is to translate the source into a target language sentence  $e_1^I = e_1 \dots e_i \dots e_I$ . The strategy is to choose the target sentence with the highest probability among all others:

$$\hat{e}_i^I = \arg \max_{I, e_1^I} \{Pr(e_1^I | f_1^J)\} \quad (1)$$

We model  $Pr(e_1^I | f_1^J)$  directly using a log-linear combination of several models (Och and Ney, 2002):

$$Pr(e_1^I | f_1^J) = \frac{\exp\left(\sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J)\right)}{\sum_{I', e_1^{I'}} \exp\left(\sum_{m=1}^M \lambda_m h_m(e_1^{I'}, f_1^J)\right)} \quad (2)$$

The denominator is to make the  $Pr(e_1^I | f_1^J)$  to be a probability distribution and it depends only on the source sentence  $f_1^J$ . For search, the decision rule is simply:

$$\hat{e}_i^I = \arg \max_{I, e_1^I} \left\{ \sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J) \right\} \quad (3)$$

The model scaling factors  $\lambda_1^M$  are trained with Minimum Error Rate Training (MERT). In this paper, the phrase-based machine translation system

is utilized (Och et al., 1999; Zens et al., 2002; Koehn et al., 2003).

## 3 Tagging-style Reordering Model

In this section, we describe the proposed novel model. First we will describe the training process. Then we explain how to use the model in the decoder.

### 3.1 Modeling

Figure 1 shows the modeling steps. The first step is word alignment training. Figure 1(a) is an example after GIZA++ training. If we regard this alignment as a translation result, i.e. given the source sentence  $f_1^7$ , the system translates it into the target sentence  $e_1^7$ , then the alignment link set  $\{a_1 = 3, a_3 = 2, a_4 = 4, a_4 = 5, a_5 = 7, a_6 = 6, a_7 = 6\}$  reveals the decoding process, i.e. the alignment implies the order in which the source words should be translated, e.g. the first generated target word  $e_1$  has no alignment, we can regard it as a translation from a NULL source word; then the second generated target word  $e_2$  is translated from  $f_3$ . We reorder the source side of the alignment to get Figure 1(b). Figure 1(b) implies the source sentence decoding sequence information, which is depicted in Figure 1(c). Using this example we describe the strategies we used for special cases in the transformation from Figure 1(b) to Figure 1(c):

- ignore the unaligned target word, e.g.  $e_1$
- the unaligned source word should follow its preceding word, the unaligned feature is kept with a \* symbol, e.g.  $f_2^*$  is after  $f_1$
- when one source word is aligned to multiple target words, only keep the alignment that links the source word to the first target word, e.g.  $f_4$  is linked to  $e_5$  and  $e_6$ , only  $f_4 - e_5$  is kept. In other words, we use this strategy to guarantee that every source word appears only once in the source decoding sequence.
- when multiple source words are aligned to one target word, put together the source words according to their original relative positions, e.g.  $e_6$  is linked to  $f_6$  and  $f_7$ . So in the decoding sequence,  $f_6$  is before  $f_7$ .

Now Figure 1(c) shows the original source sentence and its decoding sequence. By using the strategies above, it is guaranteed that the source sentence and its decoding sequence have the ex-

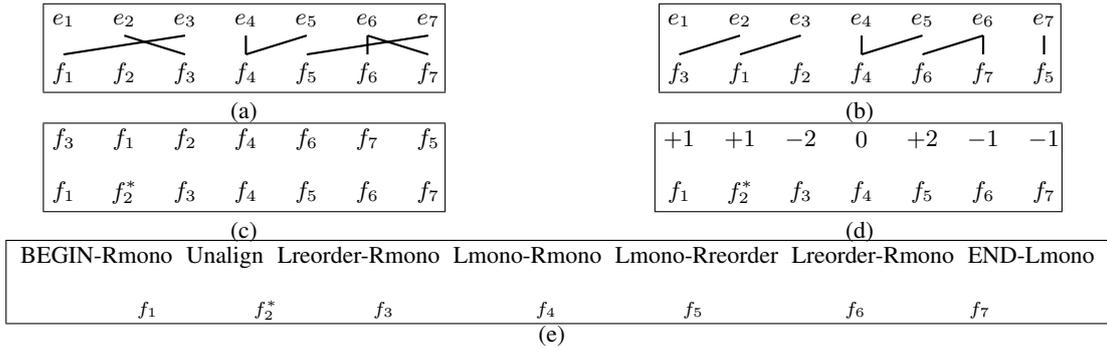


Figure 1: modeling process illustration.

actly same length. Hence the relation can be modeled by a function  $F(f)$  which assigns a value for each source word  $f$ . Figure 1(d) manifests this function. The positive function values mean that compared to the original position in the source sentence, its position in the decoding sequence should move rightwards. If the function value is 0, the word’s position in original source sentence and its decoding sequence is same. For example,  $f_1$  is the first word in the source sentence but it is the second word in the decoding sequence. So its function value is +1 (move rightwards one position).

Now Figure 1(d) converts the reordering problem into a sequence labeling or tagging problem. To make the computational cost to a reasonable level, we do a final step simplification in Figure 1(e). Suppose the longest sentence length is 100, then according to Figure 1(d), there are 200 tags (from -99 to +99 plus the unalign tag). As we will see later, this number is too large for our task. We instead design nine tags. For a source word  $f_j$  in one source sentence  $f_1^J$ , the tag of  $f_j$  will be one of the following:

- Unalign**  $f_j$  is an unaligned source word
- BEGIN-Rmono**  $j = 1$  and  $f_{j+1}$  is translated *after*  $f_j$  (Rmono for right monotonic)
- BEGIN-Rreorder**  $j = 1$  and  $f_{j+1}$  is translated *before*  $f_j$  (Rreorder for right reordered)
- END-Lmono**  $j = J$  and  $f_{j-1}$  translated *before*  $f_j$  (Lmono for left monotonic)
- END-Lreorder**  $j = J$  and  $f_{j-1}$  translated *after*  $f_j$  (Lreorder for left reordered)
- Lmono-Rmono**  $1 < j < J$  and  $f_{j-1}$  translated *before*  $f_j$  and  $f_j$  translated *before*  $f_{j+1}$
- Lreorder-Rmono**  $1 < j < J$  and  $f_{j-1}$  translated *after*  $f_j$  and  $f_j$  translated *before*  $f_{j+1}$
- Lmono-Rreorder**  $1 < j < J$  and  $f_{j-1}$  translated *before*  $f_j$  and  $f_j$  translated *after*  $f_{j+1}$
- Lreorder-Rreorder**  $1 < j < J$  and  $f_{j-1}$  trans-

lated *after*  $f_j$  and  $f_j$  translated *after*  $f_{j+1}$

Up to this point, we have converted the reordering problem into a tagging problem with nine tags. The transformation in Figure 1 is conducted for all the sentence pairs in the bilingual training corpus. After that, we have built an “annotated” corpus for the training. For this supervised learning task, we choose the approach conditional random fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006; Lavergne et al., 2010) and recurrent neural network (RNN) (Elman, 1990; Jordan, 1990; Lang et al., 1990).

For the first method, we adopt the linear-chain CRFs. However, even for the simple linear-chain CRFs, the complexity of learning and inference grows quadratically with respect to the number of output labels and the amount of structural features which are with regard to adjacent pairs of labels. Hence, to make the computational cost as low as possible, two measures have been taken. Firstly, as described above we reduce the number of tags to nine. Secondly, we add source sentence part-of-speech (POS) tags to the input. For features with window size one to three, both source words and its POS tags are used. For features with window size four and five, only POS tags are used.

As the second method, we use recurrent neural network (RNN). RNN is closely related with Multilayer Perceptrons (MLP) (Rumelhart et al., 1986), but the output of one or more hidden layers is reused as additional inputs for the network in the next time step. This structure allows the RNN to learn whole sequences without restricting itself to a fixed input window. A plain RNN has only access to the previous events in the input sequence. Hence we adopt the bidirectional RNN (BRNN) (Schuster and Paliwal, 1997) which reads the input sequence from both directions before making the prediction. The long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) is applied to

counter the effects that long distance dependencies are hard to learn with gradient descent. This is often referred to as vanishing gradient problem (Bengio et al., 1994).

### 3.2 Decoding

Once the model training is finished, we make inference on develop and test corpora which means that we get the labels of the source sentences that need to be translated. In the decoder, we add a new model which checks the labeling consistency when scoring an extended state. During the search, a sentence pair  $(f_1^J, e_1^I)$  will be formally splitted into a segmentation  $S_1^K$  which consists of  $K$  phrase pairs. Each  $s_k = (i_k; b_k, j_k)$  is a triple consisting of the last position  $i_k$  of the  $k$ th target phrase  $\tilde{e}_k$ . The start and end position of the  $k$ th source phrase  $\tilde{f}_k$  are  $b_k$  and  $j_k$ . Suppose the search state is now extended with a new phrase pair  $(\tilde{f}_k, \tilde{e}_k)$ :  $\tilde{f}_k := f_{b_k} \dots f_{j_k}$  and  $\tilde{e}_k := e_{i_{k-1}+1} \dots e_{i_k}$ . We have access to the old coverage vector, from which we know if the new phrase's left neighboring source word  $f_{b_k-1}$  and right neighboring source word  $f_{j_k+1}$  have been translated. We also have the word alignment within the new phrase pair, which is stored during the phrase extraction process. Based on the old coverage vector and alignment, we can repeat the transformation in Figure 1 to calculate the labels for the new phrase. The added model will then check the consistence between the calculated labels and the labels predicted by the reordering model. The number of source words that have inconsistent labels is the penalty and is then added into the log-linear framework as a new feature.

## 4 Comparative Study

The second part of this paper is comparative study on reordering models. Here we briefly describe those models which will be compared to later.

### 4.1 Moses lexicalized reordering model

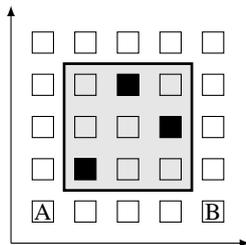


Figure 2: lexicalized reordering model illustration.

Moses (Koehn et al., 2007) contains a word-based orientation model, which has three types of reordering: (m) monotone order, (s) switch with previous phrase and (d) discontinuous. Figure 2 is an example. The definitions of reordering types are as follows:

**monotone** for current phrase, if a word alignment to the bottom left (point A) exists and there is no word alignment point at the bottom right position (point B).

**swap** for current phrase, if a word alignment to the bottom right (point B) exists and there is no word alignment point at the bottom left position (point A).

**discontinuous** all other cases

Our implementation is same with the default behavior of Moses lexicalized reordering model. We count how often each extracted phrase pair is found with each of the three reordering types. The add-0.5 smoothing is then applied. Finally, the probability is estimated with maximum likelihood principle.

### 4.2 Maximum entropy reordering model

Figure 3 is an illustration of (Zens and Ney, 2006).  $j$  is the source word position which is aligned to the last target word of the current phrase.  $j'$  is the last source word position of the current phrase.  $j''$  is the source word position which is aligned to the first target word position of the next phrase. (Zens and Ney, 2006) proposed a maximum entropy classifier to predict the orientation of the next phrase given the current phrase. The orientation class  $c_{j,j',j''}$  is defined as:

$$c_{j,j',j''} = \begin{cases} \text{left,} & \text{if } j'' < j \\ \text{right,} & \text{if } j'' > j \text{ and } j'' - j' > 1 \\ \text{monotone,} & \text{if } j'' > j \text{ and } j'' - j' = 1 \end{cases} \quad (4)$$

The orientation probability is modeled in a log-linear framework using a set of  $N$  feature functions  $h_n(f_1^J, e_1^I, i, j, c_{j,j',j''})$ ,  $n = 1, \dots, N$ . The whole model is:

$$p_{\lambda_1^N}(c_{j,j',j''} | f_1^J, e_1^I, i, j) = \frac{\exp(\sum_{n=1}^N \lambda_n h_n(f_1^J, e_1^I, i, j, c_{j,j',j''}))}{\sum_{c'} \exp(\sum_{n=1}^N \lambda_n h_n(f_1^J, e_1^I, i, j, c'))} \quad (5)$$

Different features can be used, we use the source and target word features to train the model.

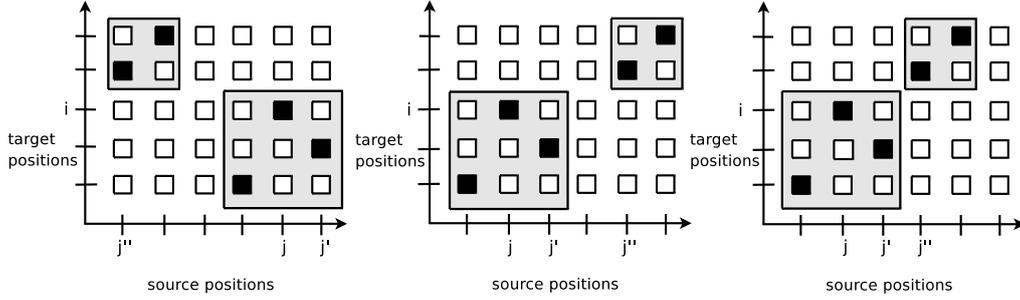


Figure 3: phrase orientation: left, right and monotone.  $j$  is the source word position aligned to the last target word of current phrase.  $j'$  is the last source word position of current phrase.  $j''$  is the source word position aligned to the first target word position of the next phrase.

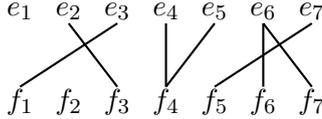


Figure 4: bilingual LM illustration. The bilingual sequence is  $e_1-$ ,  $e_2-f_3$ ,  $e_3-f_1$ ,  $e_4-f_4$ ,  $e_5-f_4$ ,  $e_6-f_6-f_7$ ,  $e_7-f_5$ .

### 4.3 Bilingual LM

The previous two models belong to “*The reordering is a classification problem*”. Now we turn to “*The reordering is a decoding order problem*”. (Mariño et al., 2006) implement a translation model using  $n$ -grams. In this way, the translation system can take full advantage of the smoothing and consistency provided by standard back-off  $n$ -gram models. Figure 4 is an example. The interpretation is that given the sentence pair  $(f_1^7, e_1^7)$  and its alignment, the correct translation order is  $e_1-$ ,  $e_2-f_3$ ,  $e_3-f_1$ ,  $e_4-f_4$ ,  $e_5-f_4$ ,  $e_6-f_6-f_7$ ,  $e_7-f_5$ . Notice the bilingual units have been ordered according to the target side, as the decoder writes the translation in a left-to-right way. Using the example we describe the strategies used for special cases:

- keep the unaligned target word, e.g.  $e_1-$
- remove the unaligned source word, e.g.  $f_2$
- when one source word aligned to multiple target words, duplicate the source word for each target word, e.g.  $e_4-f_4$ ,  $e_5-f_4$
- when multiple source words aligned to one target word, put together the source words for that target word, e.g.  $e_6-f_6-f_7$

After the operation in Figure 4 was done for all bilingual sentence pairs, we get a decoding sequence corpus. We build a 9-gram LM using SRILM toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing.

The model is added as an additional feature in Equation (2). To use the bilingual LM, the search state must be augmented to keep the bilingual unit

decoding sequence. In search, the bilingual LM is applied similar to the standard target side LM. The bilingual sequence of phrase pairs will be extracted using the same strategy in Figure 4. Suppose the search state is now extended with a new phrase pair  $(\tilde{f}, \tilde{e})$ .  $\tilde{F}$  is the bilingual sequence for the new phrase pair  $(\tilde{f}, \tilde{e})$  and  $\tilde{F}^i$  is the  $i^{\text{th}}$  unit within  $\tilde{F}$ .  $\tilde{F}'$  is the bilingual sequence history for current state. We compute the feature score  $h_{bilm}(\tilde{F}, \tilde{F}')$  of the extended state as follows:

$$h_{bilm}(\tilde{F}, \tilde{F}') = \lambda \cdot \sum_{i=1}^{|\tilde{F}|} \log p(\tilde{F}^i | \tilde{F}', \tilde{F}^1, \dots, \tilde{F}^{i-1}) \quad (6)$$

$\lambda$  is the scaling factor for this model.  $|\tilde{F}|$  is the length of the bilingual decoding sequence.

### 4.4 Source decoding sequence LM

(Feng et al., 2010) present a simpler version of the above bilingual LM where they use only the source side to model the decoding order. The source word decoding sequence in Figure 4 is then  $f_3, f_1, f_2, f_4, f_6, f_7, f_5$ . We also build a 9-gram LM based on the source word decoding sequences. The usage of the model is same as bilingual LM.

### 4.5 Syntactic cohesion model

The previous two models belong to “*The reordering is a decoding order problem*”. Now we turn to “*The reordering can be solved by outside heuristics*”. (Cherry, 2008) proposed a syntactic cohesion model. The core idea is that the syntactic structure of the source sentence should be preserved during translation. This structure is represented by a source sentence dependency tree. The algorithm is as follows: given the source sentence and its dependency tree, during the translation process, once a hypothesis is extended, check if the source dependency tree contains a subtree  $T$  such that:

- Its translation is already started (at least one node is covered)
- It is interrupted by the new added phrase (at least one word in the new source phrase is not in  $T$ )
- It is not finished (after the new phrase is added, there is still at least one free node in  $T$ )

If so, we say this hypothesis violates the subtree  $T$ , and the model returns the number of subtrees that this hypothesis violates.

#### 4.6 Semantic cohesion model

(Feng et al., 2012) propose two structure features from semantic role labeling (SRL) results. Similar to the previous model, the SRL information is used as soft constraints. During decoding process, the first feature will report how many event layers that one search state violates and the second feature will report the amount of semantic roles that one search state violates. In this paper, the two features have been used together. So when the semantic cohesion model is used, both features will be triggered.

#### 4.7 Tree-based jump model

(Wang et al., 2007) present a pre-reordering method for Chinese-English translation task. In Section 3.6 of (Zhang, 2013), instead of doing hard reordering decision, the author uses the rules as soft constraints in the decoder. In this paper, we use the similar method as described in (Zhang, 2013). Our strategy is: firstly, we parse the source sentences to get constituency trees. Then we manipulate the trees using heuristics described by (Wang et al., 2007). The leaf nodes in the revised tree constitute the reordered source sentence. Finally, in the log-linear framework (Equation 2) a new jump model is added which uses the reordered source sentence to calculate the cost. For example, the original sentence  $f_1 f_2 f_3 f_4 f_5$  is now converted by rules into the new sentence  $f_1 f_5 f_3 f_2 f_4$ . For decoding, we still use the original sentence. Suppose previously translated source phrase is  $f_1$  and the current phrase is  $f_5$ . Then the standard jump model gives cost  $q_{Dist} = 4$  and the new tree-based jump model will return a cost  $q_{Dist.new} = 1$ .

## 5 Experiments

In this section, we describe the baseline setup, the CRFs training results, the RNN training results

and translation experimental results.

### 5.1 Experimental Setup

Our baseline is a phrase-based decoder, which includes the following models: an  $n$ -gram target-side language model (LM), a phrase translation model and a word-based lexicon model. The latter two models are used for both directions:  $p(f|e)$  and  $p(e|f)$ . Additionally we use phrase count features, word and phrase penalty. The reordering model for the baseline system is the distance-based jump model which uses linear distance. This model does not have hard limit. We list the important information regarding the experimental setup below. All those conditions have been kept same in this work.

- lowercased training data from the GALE task (Table 1, UN corpus not included) alignment trained with GIZA++
- tuning corpus: NIST06  
test corpora: NIST02 03 04 05 and 08
- 5-gram LM (1 694 412 027 running words) trained by SRILM toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing training data: target side of bilingual data.
- BLEU (Papineni et al., 2001) and TER (Snover et al., 2005) reported all scores calculated in lowercase way.
- Wapiti toolkit (Lavergne et al., 2010) used for CRFs; RNN is built by the RNNLIB toolkit.

	Chinese	English
<b>Sentences</b>		5 384 856
<b>Running Words</b>	115 172 748	129 820 318
<b>Vocabulary</b>	1 125 437	739 251

Table 1: translation model and LM training data statistics

Table 1 contains the data statistics used for translation model and LM. For the reordering model, we take two further filtering steps. Firstly, we delete the sentence pairs if the source sentence length is one. When the source sentence has only one word, the translation will be always monotonic and the reordering model does not need to learn this. Secondly, we delete the sentence pairs if the source sentence contains more than three contiguous unaligned words. When this happens, the sentence pair is usually low quality hence not suitable for learning. The main purpose of the two filtering steps is to further lay down the computational burden. The label distribution is depicted in Figure 5. We can see that most words are monotonic. We then divide the corpus to three parts:

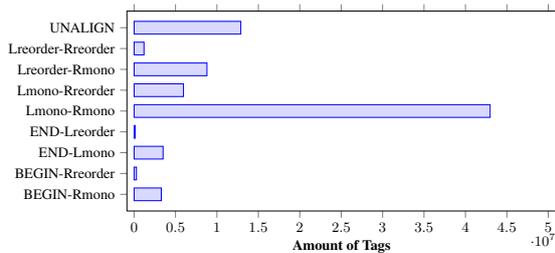


Figure 5: Tags distribution illustration.

train, validation and test. The source side data statistics for the reordering model training is given in Table 2 (target side has only nine labels).

	train	validation	test
Sentences	2 973 519	400 000	400 000
Running Words	62 263 295	8 370 361	8 382 086
Vocabulary	454 951	149686	150 007

Table 2: tagging-style model training data statistics

## 5.2 CRFs Training Results

The toolkit Wapiti (Lavergne et al., 2010) is used in this paper. We choose the classical optimization algorithm limited memory BFGS (L-BFGS) (Liu and Nocedal, 1989). For regularization, Wapiti uses both the  $\ell^1$  and  $\ell^2$  penalty terms, yielding the elastic-net penalty of the form

$$\rho_1 \cdot \|\theta\|_1 + \frac{\rho_2}{2} \cdot \|\theta\|_2^2 \quad (7)$$

In this work, we use as many features as possible because  $\ell^1$  penalty  $\rho_1 \|\theta\|_1$  is able to yield sparse parameter vectors, i.e. using a  $\ell^1$  penalty term implicitly performs the feature selection. The computational costs are given here: on a cluster with two AMD Opteron(tm) Processor 6176 (total 24 cores), the training time is about 16 hours, peak memory is around 120G. Several experiments have been done to find the suitable hyperparameter  $\rho_1$  and  $\rho_2$ , we choose the model with lowest error rate on validation corpus for translation experiments. The error rate of the chosen model on test corpus (the test corpus in Table 2) is 25.75% for token error rate and 69.39% for sequence error rate. Table 3 is the feature template we set initially which generates 722 999 637 features. Some examples are given in Table 4. After training 36 902 363 features are kept.

## 5.3 RNN Training Results

We also applied RNN to the task as an alternative approach to CRFs. The here used RNN implementation is RNNLIB which has support for long short term memory (LSTM) (Graves, 2008). We used a one of k encoding for the input word and also for the labels. After testing several configurations over the validation corpus we used a network with

Feature Templates
<b>1-gram source word features</b> $x[-4,0], x[-3,0], x[-2,0], x[-1,0]$ $x[0,0], x[1,0], x[2,0], x[3,0], x[4,0]$
<b>1-gram source POS features</b> $x[-4,1], x[-3,1], x[-2,1], x[-1,1]$ $x[0,1], x[1,1], x[2,1], x[3,1], x[4,1]$
<b>2-gram source word features</b> $x[-1,0]/x[0,0], x[0,0]/x[1,0]$ $x[-1,1]/x[0,1], x[0,1]/x[1,1]$
<b>3-gram source word features</b> $x[-1,0]/x[0,0]/x[1,0]$ $x[-2,0]/x[-1,0]/x[0,0]$ $x[0,0]/x[1,0]/x[2,0]$
<b>3-gram source POS features</b> $x[0,1]/x[1,1]/x[2,1]$ $x[-2,1]/x[-1,1]/x[0,1]$ $x[-1,1]/x[0,1]/x[1,1]$
<b>4-gram source POS features</b> $x[0,1]/x[1,1]/x[2,1]/x[3,1]$ $x[0,1]/x[-1,1]/x[-2,1]/x[-3,1]$ $x[-1,1]/x[0,1]/x[1,1]/x[2,1]$ $x[-2,1]/x[-1,1]/x[0,1]/x[1,1]$
<b>5-gram source POS features</b> $x[0,1]/x[1,1]/x[2,1]/x[3,1]/x[4,1]$ $x[-4,1]/x[-3,1]/x[-2,1]/x[-1,1]/x[0,1]$ $x[-2,1]/x[-1,1]/x[0,1]/x[1,1]/x[2,1]$
<b>bigram output label feature</b> $x[-1,2]/x[0,2]$

Table 3: feature templates for CRFs training

Words	POS	Label
基于	P	BEGIN-Rmono
这	DT	Lmono-Rmono
种	M	Lmono-Rmono
看法	NN	Lmono-Rmono
,	PU	Lmono-Rmono
本人	PN	Lmono-Rmono
是	VC	UNALIGN
支持	VV	Lmono-Rmono
修正案	NN	Lmono-Rmono
的	DEC	UNALIGN
。	PU	END-Lmono

Table 4: feature examples.  $x[\mathbf{row}, \mathbf{col}]$  specifies a token in the input data.  $\mathbf{row}$  specifies the relative position from the current label and  $\mathbf{col}$  specifies the absolute position of the column. So for the current label in this table,  $x[-1, 2]/x[0, 2]$  is Lmono-Rmono/UNALIGN and  $x[-1, 1]/x[0, 1]/x[1, 1]$  is PN/VC/VV.

LSTM 200 nodes in the hidden layer. The RNN has a token error rate of 27.31% and a sentence error rate of 77.00% over the test corpus in Table 2. The RNN is trained on a similar computer as above. RNNLIB utilizes only one thread. The training time is about three and a half days and peak memory consumption is 1G .

## 5.4 Comparison of CRFs and RNN errors

CRFs performs better than RNN (token error rate 25.75% vs 27.31%). Both error rate values are much higher than what we usually see in part-of-speech tagging task. The main reason is that the “annotated” corpus is converted from word alignment which contains lots of error. However, as we

Prediction \ Reference	Unalign	BEGIN-Rm	BEGIN-Rr	END-Lm	END-Lr	Lm-Rm	Lr-Rm	Lm-Rr	Lr-Rr
<b>Unalign</b>	687724	15084	850	7347	716	493984	107364	43457	9194
<b>BEGIN-Rmono</b>	3537	338315	6209	0	0	0	0	0	0
<b>BEGIN-Rreorder</b>	419	12557	17054	0	0	0	0	0	0
<b>END-Lmono</b>	1799	0	0	365635	3196	0	0	0	0
<b>END-Lreorder</b>	510	0	0	5239	7913	0	0	0	0
<b>Lmomo-Rmono</b>	188627	0	0	0	0	4032738	176682	150952	13114
<b>Lreorder-Rmono</b>	88177	0	0	0	0	369232	433027	27162	15275
<b>Lmomo-Rreorder</b>	32342	0	0	0	0	268570	24558	296033	10645
<b>Lreorder-Rreorder</b>	9865	0	0	0	0	34746	20382	16514	45342
Recall	50.36%	97.20%	56.79%	98.65%	57.92%	88.40%	46.42%	46.83%	35.74%
Precision	67.89%	92.45%	70.73%	96.67%	66.92%	77.56%	56.83%	55.42%	48.46%

Table 5: CRF Confusion Matrix. Abbreviations: Lmono(Lm) Lreorder(Lr) Rmono(Rm) Rreorder(Rr)

Prediction \ Reference	Unalign	BEGIN-Rm	BEGIN-Rr	END-Lm	END-Lr	Lm-Rm	Lr-Rm	Lm-Rr	Lr-Rr
<b>Unalign</b>	589100	17299	901	7870	1000	639555	82413	24277	3305
<b>BEGIN-Rmono</b>	1978	339686	6397	0	0	0	0	0	0
<b>BEGIN-Rreorder</b>	186	13812	16032	0	0	0	0	0	0
<b>END-Lmono</b>	2258	0	0	364121	4251	0	0	0	0
<b>END-Lreorde</b>	699	0	0	4693	8269	1	0	0	0
<b>Lmomo-Rmono</b>	142777	1	0	0	0	4232113	105266	78692	3264
<b>Lreorder-Rmono</b>	96278	0	1	0	0	491989	323272	14635	6698
<b>Lmomo-Rreorder</b>	31118	0	0	0	0	380483	18144	198068	4335
<b>Lreorder-Rreorder</b>	12366	0	1	0	0	50121	25196	17008	22157
Recall	43.13%	97.59%	53.39%	98.24%	60.53%	92.77%	34.65%	31.33%	17.47%
Precision	67.19%	91.61%	68.71%	96.66%	61.16%	73.04%	58.32%	59.54%	55.73%

Table 6: RNN Confusion Matrix. Abbreviations: Lmono(Lm) Lreorder(Lr) Rmono(Rm) Rreorder(Rr)

will show later, the model trained with both CRFs and RNN help to improve the translation quality.

Table 5 and Table 6 demonstrate the confusion matrix of the CRFs and RNN errors over the test corpus. The rows represent the correct tag that the classifier should have predicted and the columns are the actually predicted tags. E.g. the number 687724 in first row and first column of Table 5 tells that there are 687724 correctly labeled **Unalign** tags. The number 15084 in first row and second column of Table 5 represents that there are 15084 **Unalign** tags labeled incorrectly to **Begin-Rmono**. Therefore, numbers on the diagonal from the upper left to the lower right corner represent the amount of correctly classified tags and all other numbers show the amount of false labels. The many zeros show that both classifier rarely make mistake for the label “**BEGIN-\***” which only occur at the beginning of a sentence. The same is true for the “**END-\***” labels.

## 5.5 Translation Results

Results are summarized in Table 7. Please read the caption for the meaning of abbreviations. An **Index** column is added for score reference convenience (B for BLEU; T for TER). For the proposed model, significance testing results on both BLEU and TER are reported (B2 and B3 compared to B1, T2 and T3 compared to T1). We perform bootstrap resampling with bounds estimation as described in (Koehn, 2004). The 95% confidence threshold

(denoted by † in the table) is used to draw significance conclusions. We add a column **avg.** to show the average improvements.

From Table 7 we see that the proposed reordering model using CRFs improves the baseline by 0.98 BLEU and 1.21 TER on average, while the proposed reordering model using RNN improves the baseline by 1.32 BLEU and 1.53 TER on average. For line B2 B3 and T2 T3, most scores are better than their corresponding baseline values with more than 95% confidence. The results show that our proposed idea improves the baseline system and RNN trained model performs better than CRFs trained model, in terms of both automatic measure and significance test. To investigate why RNN has lower performance for the tagging task but achieves better BLEU, we build a 3-gram LM on the source side of the training corpus in Table 2 and perplexity values are listed in Table 8. The perplexity of the test corpus for reordering model comparison is much lower than those NIST corpora for translation experiments. In other words, there exists mismatch of the data for reordering model training and actual MT data. This could explain why CRFs is superior to RNN for labeling problem while RNN is better for MT tasks.

For the comparative study, the best method is the tree-based jump model (JUMPTREE). Our proposed model ranks the second position. The difference is tiny: on average only 0.08 BLEU (B3 and B10) and 0.15 TER (T3 and T10). Even with

Systems	NIST02	NIST03	NIST04	NIST05	NIST08	avg.	Index
<b>BLEU scores</b>							
baseline	33.60	34.29	35.73	32.15	26.34	-	B1
baseline+CRFs	34.53	35.19	36.56 $\ddagger$	33.30 $\ddagger$	27.41 $\ddagger$	0.98	B2
baseline+RNN	35.30 $\ddagger$	35.34 $\ddagger$	37.03 $\ddagger$	33.80 $\ddagger$	27.23 $\ddagger$	1.32	B3
baseline+LRM	34.87	34.90	36.40	33.43	27.45	0.99	B4
baseline+MERO	34.91	34.83	36.29	33.69	26.66	0.85	B5
baseline+BILM	35.21	35.00	36.83	33.64	27.39	1.19	B6
baseline+SRCLM	34.55	34.52	36.18	32.84	27.03	0.50	B7
baseline+SRL	35.05	34.93	36.71	33.22	26.89	0.93	B8
baseline+SC	34.96	34.52	36.37	33.35	26.90	0.79	B9
baseline+JUMPTREE	35.10	35.53	37.12	34.18	27.19	1.40	B10
baseline+LRM+MERO+BILM+SRCLM+SRL+SC+JUMPTREE	36.77	36.16	38.10	35.67	28.52	2.62	B11
baseline+LRM+MERO+BILM+SRCLM+SRL+SC+JUMPTREE+RNN	36.99	37.00	38.79	35.86	28.99	3.10	B12
<b>TER scores</b>							
baseline	61.36	60.48	59.12	60.94	65.17	-	T1
baseline+CRFs	60.14 $\ddagger$	58.91 $\ddagger$	57.91 $\ddagger$	59.77 $\ddagger$	64.30 $\ddagger$	1.21	T2
baseline+RNN	59.38 $\ddagger$	58.87 $\ddagger$	57.60 $\ddagger$	59.56 $\ddagger$	63.99 $\ddagger$	1.53	T3
baseline+LRM	60.07	59.08	58.42	59.74	64.50	1.05	T4
baseline+MERO	60.19	59.58	58.51	59.49	64.68	0.92	T5
baseline+BILM	60.23	59.93	58.59	60.09	64.72	0.70	T6
baseline+SRCLM	60.27	59.55	58.40	60.16	64.61	0.82	T7
baseline+SRL	60.05	59.55	58.14	59.69	64.74	0.98	T8
baseline+SC	59.90	59.37	58.27	59.69	64.44	1.08	T9
baseline+JUMPTREE	59.53	58.54	57.67	58.90	64.04	1.68	T10
baseline+LRM+MERO+BILM+SRCLM+SRL+SC+JUMPTREE	59.16	57.84	56.83	58.03	63.20	2.40	T11
baseline+LRM+MERO+BILM+SRCLM+SRL+SC+JUMPTREE+RNN	58.67	57.67	56.27	58.00	63.09	2.67	T12

Table 7: Experimental results. CRFs and RNN mean the tagging-style model trained with CRFs or RNN; LRM for lexicalized reordering model (Koehn et al., 2007); MERO for maximum entropy reordering model (Zens and Ney, 2006); BILM for bilingual language model (Mariño et al., 2006) and SRCLM for its simpler version source decoding sequence model (Feng et al., 2010); SC for syntactic cohesion model (Cherry, 2008); SRL for semantic cohesion model (Feng et al., 2012); JUMPTREE for our tree-based jump model based on (Wang et al., 2007).

Test in Table 2	Running Words	OOV	Perplexity
NIST02	8 382 086	33854	74.364
NIST03	22 749	195	176.806
NIST04	24 180	290	274.679
NIST05	49 612	320	170.507
NIST08	29 966	228	279.402
NIST08	32 502	511	408.067

Table 8: perplexity

a strong system (B11 and T11), our model is still able to provide improvements (B12 and T12).

## 6 Conclusion

In this paper, a novel tagging style reordering model has been proposed. By our method, the reordering problem is converted into a sequence labeling problem so that the whole source sentence is taken into consideration for reordering decision. By adding an unaligned word tag, the unaligned word phenomenon is automatically implanted in the proposed model. The model is utilized as soft constraints in the decoder. In practice, we do not experience decoding memory increase nor speed slow down.

We choose CRFs and RNN to accomplish the sequence labeling task. The CRFs achieves lower error rate on the tagging task but RNN trained model is better for the translation task. Experimental results show that our model is stable and improves the baseline system by 0.98 BLEU and 1.21 TER (trained by CRFs) and 1.32 BLEU and 1.53 TER (trained by RNN). Most of the scores

are better than their corresponding baseline values with more than 95% confidence. We also compare our method with several other popular reordering models. Our model ranks the second position which is slightly worse than the tree-based jump model. However, the tree-based jump model relies on manually designed reordering rules which does not exist for many language pairs while our model can be easily adapted to other translation tasks. We also show that the proposed model is able to improve a very strong baseline system.

The main contributions of the paper are: propose the tagging-style reordering model and improve the translation quality; compare two sequence labeling techniques CRFs and RNN; compare our method with seven other reordering models. To our best knowledge, it is the first time that the above two comparisons have been reported .

## Acknowledgments

This work was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation, and also partly based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-12-C-0015. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).

## References

- Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, March.
- Adam Berger, Peter F. Brown, Stephen A. Pietra, Vincent J. Pietra, Andrew S. Kehler, and Robert L. Mercer. 1996. Language translation apparatus and method of using Context-Based translation models. *United States Patent*, No. 5,510,981.
- Colin Cherry. 2008. Cohesive phrase-based decoding for statistical machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL: HLT)*, pages 72–80, Columbus, Ohio, USA, June. Association for Computational Linguistics.
- Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.
- Minwei Feng, Arne Mauser, and Hermann Ney. 2010. A source-side decoding sequence model for statistical machine translation. In *Proceedings of the Association for Machine Translation in the Americas (AMTA)*, Denver, CO, USA, October.
- Minwei Feng, Weiwei Sun, and Hermann Ney. 2012. Semantic cohesion model for phrase-based SMT. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 867–878, Mumbai, India, December. The COLING 2012 Organizing Committee.
- Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 848–856, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Alex Graves. 2008. *Supervised Sequence Labelling with Recurrent Neural Networks*. Ph.D. thesis, Technical University of Munich, July.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November.
- Michael I. Jordan. 1990. Attractor dynamics and parallelism in a connectionist sequential machine. *IEEE Computer Society Neural Networks Technology Series*, pages 112–127.
- Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL) - Volume 1*, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session*, pages 177–180, Prague, Czech Republic, June.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Kevin J. Lang, Alex H. Waibel, and Geoffrey E. Hinton. 1990. A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43, January.
- Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513. Association for Computational Linguistics, July.
- Dong C. Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(3):503–528, December.
- José B. Mariño, Rafael E. Banchs, Josep Maria Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-Jussà. 2006. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, December.
- Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, Pennsylvania, USA, July.
- Franz J. Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP)*, pages 20–28, University of Maryland, College Park, MD, USA, June. Association for Computational Linguistics.

- Franz J. Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proceedings of the Conference on Statistical Machine Translation at the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT)*, pages 161–168, Boston, Massachusetts, USA, May. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. *IBM Research Report*, RC22176 (W0109-022), September.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning internal representations by error propagation. In David E. Rumelhart and James L. McClelland, editors, *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*, pages 318–362. MIT Press, Cambridge, MA, USA.
- Mike Schuster and Kuldeep K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, November.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, John Makhoul, Linnea Micciulla, and Ralph Weischedel. 2005. A study of translation error rate with targeted human annotation. Technical Report LAMP-TR-126, CS-TR-4755, UMIACS-TR-2005-58, University of Maryland, College Park, MD.
- Andreas Stolcke. 2002. Srilm - an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing (ICSLP)*, pages 901–904, Denver, Colorado, USA, September. ISCA.
- Charles Sutton and Andrew McCallum, 2006. *Introduction to Conditional Random Fields for Relational Learning*, pages 93–128. MIT Press.
- Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 737–745, Prague, Czech Republic, June. Association for Computational Linguistics.
- Dekai Wu. 1995. Stochastic inversion transduction grammars with application to segmentation, bracketing, and alignment of parallel corpora. In *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI) - Volume 2*, pages 1328–1335, San Francisco, CA, USA, August. Morgan Kaufmann Publishers Inc.
- Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL) - Volume 1*, pages 144–151, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Richard Zens and Hermann Ney. 2006. Discriminative reordering models for statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation at the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT)*, pages 55–63, New York City, NY, June. Association for Computational Linguistics.
- Richard Zens, Franz J. Och, and Hermann Ney. 2002. Phrase-based statistical machine translation. In *German Conference on Artificial Intelligence*, pages 18–32. Springer Verlag, September.
- Yuqi Zhang, Richard Zens, and Hermann Ney. 2007. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of the Workshop on Syntax and Structure in Statistical Translation at the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL-HLT)/Association for Machine Translation in the Americas (AMTA)*, pages 1–8, Morristown, NJ, USA, April. Association for Computational Linguistics.
- Yuqi Zhang. 2013. *The Application of Source Language Information in Chinese-English Statistical Machine Translation*. Ph.D. thesis, Computer Science Department, RWTH Aachen University, May.

# A Markov Model of Machine Translation using Non-parametric Bayesian Inference

Yang Feng and Trevor Cohn

Department of Computer Science

The University of Sheffield

Sheffield, United Kingdom

yangfeng145@gmail.com and t.cohn@sheffield.ac.uk

## Abstract

Most modern machine translation systems use phrase pairs as translation units, allowing for accurate modelling of phrase-internal translation and reordering. However phrase-based approaches are much less able to model sentence level effects between different phrase-pairs. We propose a new model to address this imbalance, based on a word-based Markov model of translation which generates target translations left-to-right. Our model encodes word and phrase level phenomena by conditioning translation decisions on previous decisions and uses a hierarchical Pitman-Yor Process prior to provide dynamic adaptive smoothing. This mechanism implicitly supports not only traditional phrase pairs, but also gapping phrases which are non-consecutive in the source. Our experiments on Chinese to English and Arabic to English translation show consistent improvements over competitive baselines, of up to +3.4 BLEU.

## 1 Introduction

Recent years have witnessed burgeoning development of statistical machine translation research, notably phrase-based (Koehn et al., 2003) and syntax-based approaches (Chiang, 2005; Galley et al., 2006; Liu et al., 2006). These approaches model sentence translation as a sequence of simple translation decisions, such as the application of a phrase translation in phrase-based methods or a grammar rule in syntax-based approaches. In order to simplify modelling, most MT models make an independence assumption, stating that the translation decisions in a derivation are independent of one another. This conflicts with the intuition behind phrase-based MT, namely that translation decisions should be dependent on con-

text. On one hand, the use of phrases can memorize local context and hence helps to generate better translation compared to word-based models (Brown et al., 1993; Och and Ney, 2003). On the other hand, this mechanism requires each phrase to be matched strictly and to be used as a whole, which precludes the use of discontinuous phrases and leads to poor generalisation to unseen data (where large phrases tend not to match).

In this paper we propose a new model to drop the independence assumption, by instead modelling correlations between translation decisions, which we use to induce translation derivations from aligned sentences (akin to word alignment). We develop a Markov model over translation decisions, in which each decision is conditioned on previous  $n$  most recent decisions. Our approach employs a sophisticated Bayesian non-parametric prior, namely the hierarchical Pitman-Yor Process (Teh, 2006; Teh et al., 2006) to represent back-off from larger to smaller contexts. As a result, we need only use very simple translation units – primarily single words, but can still describe complex multi-word units through correlations between their component translation decisions. We further decompose the process of generating each target word into component factors: finishing the translating, jumping elsewhere in the source, emitting a target word and deciding the fertility of the source words.

Overall our model has the following features:

1. enabling model parameters to be shared between similar translation decisions, thereby obtaining more reliable statistics and generalizing better from small training sets.
2. learning a much richer set of translation fragments, such as *gapping* phrases, e.g., the translation for the German *werde ... ankommen* in English is *will arrive ...*
3. providing a unifying framework spanning word-based and phrase-based model of translation, while incorporating explicit transla-

tion, insertion, deletion and reordering components.

We demonstrate our model on Chinese-English and Arabic-English translation datasets. The model produces uniformly better translations than those of a competitive phrase-based baseline, amounting to an improvement of up to 3.4 BLEU points absolute.

## 2 Related Work

Word based models have a long history in machine translation, starting with the venerable IBM translation models (Brown et al., 1993) and the hidden Markov model (Vogel et al., 1996). These models are still in wide-spread use today, albeit only as a preprocessing step for inferring word level alignments from sentence-aligned parallel corpora. They combine a number of factors, including distortion and fertility, which have been shown to improve word-alignment and translation performance over simpler models. Our approach is similar to these works, as we also develop a word-based model, and explicitly consider similar translation decisions, alignment jumps and fertility. We extend these works in two important respects: 1) while they assume a simple parameterisation by making *iid* assumptions about each translation factor, we instead allow for rich correlations by modelling sequences of translation decisions; and 2) we develop our model in the Bayesian framework, using a hierarchical Pitman-Yor Process prior with rich backoff semantics between high and lower order sequences of translation decisions. Together this results in a model with rich expressiveness but can still generalize well to unseen data.

More recently, a number of authors have proposed Markov models for machine translation. Vaswani et al. (2011) propose a rule Markov model for a tree-to-string model which models correlations between pairs of minimal rules, and use Kneser-Ney smoothing to alleviate the problems of data sparsity. Similarly, Crego et al. (2011) develop a bilingual language model which incorporates words in the source and target languages to predict the next unit, which they use as a feature in a translation system. This line of work was extended by Le et al. (2012) who develop a novel estimation algorithm based around discriminative projection into continuous spaces. Also relevant is Durrani et al. (2011), who present a sequence model of translation including reordering.

Our work also uses bilingual information, using the source words as part of the conditioning context. In contrast to these approaches which primarily address the decoding problem, we focus on the learning problem of inferring alignments from parallel sentences. Additionally, we develop a full generative model using a Bayesian prior, and incorporate additional factors besides lexical items, namely jumps in the source and word fertility.

Another aspect of this paper is the implicit support for phrase-pairs that are discontinuous in the source language. This idea has been developed explicitly in a number of previous approaches, in grammar based (Chiang, 2005) and phrase-based systems (Galley and Manning, 2010). The latter is most similar to this paper, and shows that discontinuous phrases compliment standard contiguous phrases, improving expressiveness and translation performance. Unlike their work, here we develop a complimentary approach by constructing a generative model which can induce these rich rules directly from sentence-aligned corpora.

## 3 Model

Given a source sentence, our model infers a latent derivation which produces a target translation and meanwhile gives a word alignment between the source and the target. We consider a process in which the target string is generated using a left-to-right order, similar to the decoding strategy used by phrase-based machine translation systems (Koehn et al., 2003). During this process we maintain a position in the source sentence, which can jump around to allow for different sentence ordering in the target vs. source languages. In contrast to phrase-based models, we use words as our basic translation unit, rather than multi-word phrases. Furthermore, we decompose the decisions involved in generating each target word to a number of separate factors, where each factor is modelled separately and conditioned on a rich history of recent translation decisions.

### 3.1 Markov Translation

Our model generates target translation left-to-right word by word. The generative process employs the following recursive procedure to construct the target sentence conditioned on the source:

```
 $i \leftarrow 1$   
while Not finished do  
    Decide whether to finish the translation,  $\xi_i$ 
```

Step	Source sentence	Translation	<i>finish</i>	<i>jump</i>	<i>emission</i>
0	— Je le prends				
1	— <u>Je</u> le prends	I	<i>no</i>	<i>monotone</i>	Je → I
2	— <u>Je</u> le prends	I 'll	<i>no</i>	<i>insert</i>	null → 'll
3	— <u>Je</u> le prends	I 'll take	<i>no</i>	<i>forward</i>	prends → take
4	— Je <u>le</u> prends	I 'll take that	<i>no</i>	<i>backward</i>	le → that
5	— Je <u>le</u> prends	I 'll take that one	<i>no</i>	<i>stay</i>	le → one
6	— Je <u>le</u> prends	I 'll take that one	<i>yes</i>		

Figure 1: Translation agenda of Je le prends → I 'll take that one.

**if**  $\xi_i = \text{false}$  **then**  
 Select a source word to jump to  
 Emit a target word for the source word  
**end if**  
 $i \leftarrow i + 1$   
**end while**

In the generation of each target word, our model includes three separate factors: the binary *finish* decision, a *jump* decision to move to a different source word, and *emission* which translates or otherwise inserts a word in the target string. This generative process resembles the sequence of translation decisions considered by a standard MT decoder (Koehn et al., 2003), but note that our approach differs in that there is no constraint that all words are translated exactly once. Instead source words can be skipped or repeatedly translated. This makes the approach more suitable for learning alignments, e.g., to account for word fertilities (see §3.3), while also permitting inference using Gibbs sampling (§4).

More formally, we can express our probabilistic model as

$$\begin{aligned}
 p_{bs}(e_1^I, a_1^I | f_1^J) &= \prod_{i=1}^{I+1} p(\xi_i | f_{a_i-n}^{i-1}, e_{i-n}^{i-1}) \\
 &\times \prod_{i=1}^I p(\tau_i | f_{a_i-n}^{i-1}, e_{i-n}^{i-1}) \\
 &\times \prod_{i=1}^I p(e_i | \tau_i, f_{a_i-n}^i, e_{i-n}^{i-1}) \quad (1)
 \end{aligned}$$

where  $\xi_i$  is the finish decision for target position  $i$ ,  $\tau_i$  is the jump decision to source word  $f_{a_i}$  and  $f_{a_i-n}^i$  is the source words for target positions  $i-n, i-n+1, \dots, i$ . Each of the three distributions (finish, jump and emission) is drawn respectively from hierarchical Pitman-Yor Process priors, as described in Section 3.2.

The jump decision  $\tau_i$  in Equation 1 demands further explanation. Instead of modelling jump distances explicitly, which poses problems for

generalizing between different lengths of sentences and general parameter explosion, we consider a small handful of types of jump based on the distance between the current source word  $a_i$  and the previous source word  $a_{i-1}$ , i.e.,  $d_i = a_i - a_{i-1}$ .<sup>1</sup> We bin jumps into five types:

- insert*;
- backward*, if  $d_i < 0$ ;
- stay*, if  $d_i = 0$ ;
- monotone*, if  $d_i = 1$ ;
- forward*, if  $d_i > 1$ .

The special jump type *insert* handles null alignments, denoted  $a_i = 0$  which licence spurious insertions in the target string.

To illustrate this translation process, Figure 1 shows the example translation  $\langle Je\ le\ prends,\ I\ 'll\ take\ that\ one \rangle$ . Initially we set the source position before the first source word *Je*. Then in step 1, we decide not to finish (finish=*no*), jump to source word *Je* and translate it as *I*. Next, we again decide not to finish, jump to the *null* source word and insert *'ll*. The process continues until in step 6 we elect to finish (finish=*yes*), at which point the translation is complete, with target string *I 'll take that one*.

### 3.2 Hierarchical Pitman-Yor Process

The Markov assumption limits the context of each distribution to the  $n$  most recent translation decisions, which limits the number of model parameters. However for any non-trivial value  $n > 0$ , overfitting is a serious concern. We counter the problem of a large parameter space using a Bayesian non-parametric prior, namely the hierarchical Pitman-Yor Process (PYP). The PYP describes distributions over possibly infinite event spaces that follow a power law, with few events taking the majority of the probability mass and a long tail of less frequent events. We consider a hierarchical PYP, where a sequence of chained PYP

<sup>1</sup>For a target position aligned to null, we denote its source word as *null* and set its aligned source position as that of the previous target word that is aligned to non-null.

priors allow backoff from larger to smaller contexts such that our model can learn rich contextual models for known (large) contexts while also still being able to generalize well to unseen contexts (using smaller histories).

### 3.2.1 Pitman-Yor Process

A PYP (Pitman and Yor, 1997) is defined by its discount parameter  $0 \leq a < 1$ , strength parameter  $b > -a$  and base distribution  $G_0$ . For a distribution drawn from a PYP,  $G \sim \mathcal{PYP}(a, b, G_0)$ , marginalising out  $G$  leads to a simple distribution which can be described using a variant of the Chinese Restaurant Process (CRP). In this analogy we imagine a restaurant has an infinite number of tables and each table can accommodate an infinite number of customers. Each customer (a sample from  $G$ ) walks in one at a time and seats themselves at a table. Finally each table is served a communal dish (a draw from  $G_0$ ), which is served to each customer seated at the table. The assignment of customers to tables is such that popular tables are more likely to be chosen, and this rich-get-richer dynamic produces power-law distributions with few events (the dishes at popular tables) dominating the distribution.

More formally, at time  $n$  a customer enters and selects a table  $k$  which is either a table having been seated ( $1 \leq k \leq K^-$ ) or an empty table ( $k = K^- + 1$ ) by

$$p(t_n = k | \mathbf{t}_{-n}) = \begin{cases} \frac{c_{\mathbf{t}_k}^- - a}{n-1+b} & 1 \leq k \leq K^- \\ \frac{aK^- + b}{n-1+b} & k = K^- + 1 \end{cases}$$

where  $t_n$  is the table selected by the customer  $n$ ,  $\mathbf{t}_{-n}$  is the seating arrangement of previous  $n-1$  customers,  $c_{\mathbf{t}_k}^-$  is the number of customers seated at table  $k$  in  $\mathbf{t}_{-n}$  and  $K^- = K(\mathbf{t}_{-n})$  is the number of tables in  $\mathbf{t}_{-n}$ .

If the customer sits at an empty table, a dish  $h$  is served to his table by the probability of  $G_0(h)$ , otherwise, he can only share with others the dish having been served to his table.<sup>2</sup> Overall, the probability of the customer being served a dish  $h$  is

$$p(o_n = h | \mathbf{t}_{-n}, \mathbf{o}_{-n}) = \frac{c_{\mathbf{o}_h}^- - aK_h^-}{n-1+b} + \frac{(aK^- + b)}{n-1+b} G_0(h)$$

where  $o_n$  is the dish served to the customer  $n$ ,  $\mathbf{o}_{-n}$  is the dish accommodation of previous  $n-1$  customers,  $c_{\mathbf{o}_h}^-$  is the number of customers who are

<sup>2</sup>We also say the customer is served with this dish.

served with the dish  $h$  in  $\mathbf{o}_{-n}$  and  $K_h^-$  is the number of tables served with the dish  $h$  in  $\mathbf{t}_{-n}$ .

The hierarchical PYP (hPYP; Teh (2006)) is an extension of the PYP in which the base distribution  $G_0$  is itself a PYP distribution. This parent (base) distribution can itself have a PYP as a base distribution, giving rise to hierarchies of arbitrary depth. Like the PYP, inference under the hPYP can be also described in terms of CRP whereby each table in one restaurant corresponds to a dish in the next deeper level, and is said to share the same dish. Whenever an empty table is seated in one level, a customer must enter the restaurant in the next deeper level and find a table to sit. This process continues until the customer is assigned a shared table or the deepest level of the hierarchy is reached. A similar process occurs when a customer leaves, where newly emptied tables must be propagated up the hierarchy in the form of departing customers. There is not space for a complete treatment of the hPYP and the particulars of inference; we refer the interested reader to Teh (2006).

### 3.2.2 A Hierarchical PYP Translation Model

We draw the distributions for the various translation factors from respective hierarchical PYP priors, as shown in Figure 2 for the finish, jump and emission factors. For the emission factor (Figure 2c), we draw the target word  $e_i$  from a distribution conditioned on the last two source and target words, as well as the current source word,  $f_{a_i}$  and the current jump type  $\tau_i$ . Here the draw of a target word corresponds to a customer entering and which target word to emit corresponds to which dish to be served to the customer in the CRP. The hierarchical prior encodes a backoff path in which the jump type is dropped first, followed by pairs of source and target words from least recent to most recent. The final backoff stages drop the current source word, terminating with the uniform base distribution over the target vocabulary  $V$ .

The distributions over the other two factors in Figure 2 follow a similar pattern. Note however that these distributions don't condition on the current source word, and consequently have fewer levels of backoff. The terminating base distribution for the finish factor is a uniform distribution with equal probability for finishing versus continuing. The jump factor has an additional conditioning variable  $t$  which encodes whether the previous alignment is near the start or end of the source sentence. This information affects which of the jump values are legal from the current position, such

$$\begin{array}{lll}
\xi_i | f_{a^{i-2}}^{i-1}, e_{i-2}^{i-1} \sim G_{f_{a^{i-2}, e_{i-2}^{i-1}}}^\xi & \tau_i | f_{a^{i-2}}^{i-1}, e_{i-2}^{i-1}, t \sim G_{f_{a^{i-2}, e_{i-2}^{i-1}}, t}^\tau & e_i | \tau_i, f_{a^{i-2}}^{i-1}, e_{i-2}^{i-1} \sim G_{\tau_i, f_{a^{i-2}, e_{i-2}^{i-1}}}^e \\
G_{f_{a^{i-2}, e_{i-2}^{i-1}}}^\xi \sim \mathcal{PYP}(a_3^\xi, b_3^\xi, G_{f_{a^{i-1}, e_{i-1}^{i-1}}}^\xi) & G_{f_{a^{i-2}, e_{i-2}^{i-1}}, t}^\tau \sim \mathcal{PYP}(a_3^\tau, b_3^\tau, G_{f_{a^{i-1}, e_{i-1}^{i-1}}, t}^\tau) & G_{\tau_i, f_{a^{i-2}, e_{i-2}^{i-1}}}^e \sim \mathcal{PYP}(a_5^e, b_5^e, G_{f_{a^{i-2}, e_{i-2}^{i-1}}}^e) \\
G_{f_{a^{i-1}, e_{i-1}^{i-1}}}^\xi \sim \mathcal{PYP}(a_2^\xi, b_2^\xi, G^\xi) & G_{f_{a^{i-1}, e_{i-1}^{i-1}}, t}^\tau \sim \mathcal{PYP}(a_2^\tau, b_2^\tau, G_t^\tau) & G_{f_{a^{i-2}, e_{i-2}^{i-1}}}^e \sim \mathcal{PYP}(a_4^e, b_4^e, G_{f_{a^{i-1}, e_{i-1}^{i-1}}}^e) \\
G^\xi \sim \mathcal{PYP}(a_1^\xi, b_1^\xi, G_0^\xi) & G_t^\tau \sim \mathcal{PYP}(a_1^\tau, b_1^\tau, G_{0,t}^\tau) & G_{f_{a^{i-1}, e_{i-1}^{i-1}}}^e \sim \mathcal{PYP}(a_3^e, b_3^e, G_{f_{a^i}^e}) \\
G_0^\xi \sim \mathcal{U}(\frac{1}{2}) & G_{0,t}^\tau \sim \mathcal{U} & G_{f_{a^i}^e} \sim \mathcal{PYP}(a_2^e, b_2^e, G^e) \\
& & G^e \sim \mathcal{PYP}(a_1^e, b_1^e, G_0^e) \\
& & G_0^e \sim \mathcal{U}(\frac{1}{|V|})
\end{array}$$

(a) Finish factor
(b) Jump factor
(c) Emission factor

Figure 2: Distributions over the translation factors and their hierarchical priors.

that a jump could not go outside the bounds of the source sentence. Accordingly we maintain separate distributions for each setting, and each has a different uniform base distribution parameterized according to the number of possible jump types.

### 3.3 Fertility

For each target position, our Markov model may select a source word which has been covered, which means a source word may be linked to several target positions. Therefore, we introduce *fertility* to denote the number of target positions a source word is linked to in a sentence pair. Brown et al. (1993) have demonstrated the usefulness of fertility in probability estimation: IBM models 3–5 exhibit large improvements over models 1–2. On these grounds, we include fertility to produce our *advanced* model,

$$p_{ad}(e_1^I, a_1^I | f_1^J) = p_{bs}(e_1^I, a_1^I | f_1^J) \prod_{j=1}^J p(\phi_j | f_{j-n}^j) \quad (2)$$

where  $\phi_j$  is the fertility of source word  $f_j$  in the sentence pair  $\langle f_1^J, e_1^I \rangle$  and  $p_{bs}$  is the basic model defined in Eq. 1. In order to avoid problems of data sparsity, we bin fertility into three types, a) *zero*, if  $\phi = 0$ ; b) *single*, if  $\phi = 1$ ; and c) *multiple*, if  $\phi > 1$ .

We draw the fertility variables from a hierarchical PYP distribution, using three levels of backoff,

$$\begin{aligned}
\phi_j | f_{j-1}^j &\sim G_{f_{j-1}^j}^\phi \\
G_{f_{j-1}^j}^\phi &\sim \mathcal{PYP}(a_3^\phi, b_3^\phi, G_{f_j}^\phi) \\
G_{f_j}^\phi &\sim \mathcal{PYP}(a_2^\phi, b_2^\phi, G^\phi) \\
G^\phi &\sim \mathcal{PYP}(a_1^\phi, b_1^\phi, G_0^\phi) \\
G_0^\phi &\sim \mathcal{U}(\frac{1}{3})
\end{aligned}$$

where we condition the fertility of each word token on the token to its left, which we drop during the first stage of backoff to simple word-based fertility. The last level of backoff further generalises to a shared fertility across all words. In this way we gain the benefits of local context on fertility, while including more general levels to allow wider applicability.

## 4 Gibbs Sampling

To train the model, we use Gibbs sampling, a Markov Chain Monte Carlo (MCMC) technique for posterior inference. Specifically we seek to infer the latent sequence of translation decisions given a corpus of sentence pairs. Given the structure of our model, a word alignment uniquely specifies the translation decisions and the sequence follows the order of the target sentence left to right. Our Gibbs sampler operates by sampling an update to the alignment of each target word in the corpus. It visits each sentence pair in the corpus in a random order and resamples the alignments for each target position as follows. First we discard the alignment to the current target word and decrement the counts of all factors affected by this alignment in their top level distributions (which will percolate down to the lower restaurants). Next we calculate posterior probabilities for all possible alignment to this target word based on the table occupancies in the hPYP. Finally we draw an alignment and increment the table counts for the translation decisions affected by the new alignment.

More specifically, we consider sampling from Equation 2 with  $n = 2$ . When changing the alignment to a target word  $e_i$  from  $j'$  to  $j$ , the *finish*, *jump* and *emission* for three target positions  $i, i + 1, i + 2$  and *fertility* for two source positions  $j, j'$  may be affected. This leads to the following

decrement	increment
$\xi(\text{no} \mid \text{null}, \text{'ll}, \text{Je}, \text{I})$	$\xi(\text{no} \mid \text{null}, \text{'ll}, \text{Je}, \text{I})$
$\xi(\text{no} \mid \text{p..s}, \text{take}, \text{null}, \text{'ll})$	$\xi(\text{no} \mid \text{Je}, \text{take}, \text{null}, \text{'ll})$
$\xi(\text{no} \mid \text{le}, \text{that}, \text{p..s}, \text{take})$	$\xi(\text{no} \mid \text{le}, \text{that}, \text{Je}, \text{take})$
$\tau(f \mid \text{null}, \text{'ll}, \text{Je}, \text{I})$	$\tau(s \mid \text{null}, \text{'ll}, \text{Je}, \text{I})$
$\tau(b \mid \text{p..s}, \text{take}, \text{null}, \text{'ll})$	$\tau(m \mid \text{Je}, \text{take}, \text{null}, \text{'ll})$
$\tau(s \mid \text{le}, \text{that}, \text{p..s}, \text{take})$	$\tau(s \mid \text{le}, \text{that}, \text{Je}, \text{take})$
$e(\text{take} \mid f, \text{p..s}, \text{null}, \text{'ll}, \text{Je}, \text{I})$	$e(\text{take} \mid s, \text{Je}, \text{null}, \text{'ll}, \text{Je}, \text{I})$
$e(\text{that} \mid b, \text{le}, \text{p..s}, \text{take}, \text{null}, \text{'ll})$	$e(\text{that} \mid m, \text{le}, \text{Je}, \text{take}, \text{null}, \text{'ll})$
$e(\text{one} \mid s, \text{le}, \text{le}, \text{that}, \text{p..s}, \text{take})$	$e(\text{one} \mid s, \text{le}, \text{le}, \text{that}, \text{Je}, \text{take})$
$\phi(\text{single} \mid \text{p..s}, \text{le})$	$\phi(\text{multiple} \mid \text{Je}, \langle s \rangle)$

Table 1: The count update when changing the aligned source word of take from prends to Je in Figure 1. Key: f–forward s–stay b–backward m–monotone p..s–prends.

posterior probability

$$p(a_i = j \mid \mathbf{t}_{-i}, \mathbf{o}_{-i}) \propto \prod_{l=i}^{i+2} p(\xi_l) p(\tau_l) p(e_l) \times \frac{p(\phi_j + 1) p(\phi_{j'} - 1)}{p(\phi_j) p(\phi_{j'})} \quad (3)$$

where  $\phi_j, \phi_{j'}$  are the fertilities before changing the link and for brevity we omit the conditioning contexts. For example, in Figure 1, we sample for target word *take* and change the aligned source word from *prends* to *Je*, then the items for which we need to decrement and increment the counts by one are shown in Table 1 and the posterior probability corresponding to the new alignment is the product of the hierarchical PYP probabilities of all increment items divided by the probability of the fertility of *prends* being *single*.

Maintaining the current state of the hPYP as events are incremented and decremented is non-trivial and the naive approach requires significant book-keeping and has poor runtime behaviour. For this we adopt the approach of Blunsom et al. (2009b), who present a method for maintaining table counts without needing to record the table assignments for each translation decision. Briefly, this algorithm samples the table assignment during the increment and decrement operations, which is then used to maintain aggregate table statistics. This can be done efficiently and without the need for explicit table assignment tracking.

#### 4.1 Hyperparameter Inference

In our model, we treat all hyper-parameters  $\{(a^x, b^x), x \in (\xi, \tau, e, \phi)\}$  as latent random variables rather than fixed parameters. This means our model is parameter free, and requires no user intervention when adapting to different data sets. For

the discount parameter, we employ a uniform Beta distribution  $a^x \sim \text{Beta}(1, 1)$  while for the strength parameter, we employ a vague Gamma distribution  $b^x \sim \text{Gamma}(10, 0.1)$ . All restaurants in the same level share the same hyper-prior and the hyper-parameters for all levels are resampled using slice sampling (Johnson and Goldwater, 2009) every 10 iterations.

#### 4.2 Parallel Implementation

As mentioned above, the hierarchical PYP takes into consideration a rich history to evaluate the probabilities of translation decisions. But this leads to difficulties when applying the model to large data sets, particularly in terms of tracking the table and customer counts. We apply the technique from Blunsom et al. (2009a) of using multiple processors to perform approximate Gibbs sampling which they showed achieved equivalent performance to the exact Gibbs sampler. Each process performs sampling on a subset of the corpus using local counts, and communicates changes to these counts after each full iteration. All the count deltas are then aggregated by each process to refresh the counts at the end of each iteration. In this way each process uses slightly “out-of-date” counts, but can process the data independently of the other processes. We found that this approximation improved the runtime significantly with no noticeable effect on accuracy.

### 5 Experiments

In principle our model could be directly used as a MT decoder or as a feature in a decoder. However in this paper we limit our focus to inducing word alignments, i.e., by using the model to infer alignments which are then used in a standard phrase-based translation pipeline. We leave full decoding for later work, which we anticipate would further improve performance by exploiting gapping phrases and other phenomena that implicitly form part of our model but are not represented in the phrase-based decoder. Decoding under our model would be straight-forward in principle, as the generative process was designed to closely parallel the search procedure in the phrase-based model.<sup>3</sup>

Three data sets were used in the experiments: two Chinese to English data sets on small (IWSLT) and larger corpora (FBIS), and Arabic

<sup>3</sup>However the reverse translation probability would be intractable, as this does not decompose following a left-to-right generation order in the target language.

to English translation. Our experiments seek to test how the model compares to a GIZA++ baseline, quantifies the effect of each factor in the probabilistic model (i.e., jump, fertility), and the effect of different initialisations of the sampler. We present results on translation quality and word alignment.

## 5.1 Data Setup

The Markov order of our model in all experiments was set to  $n = 2$ , as shown in Equation 2. For each data set, Gibbs sampling was performed on the training set in each direction (source-to-target and target-to-source), initialized using GIZA++.<sup>4</sup> We used the *grow* heuristic to combine the GIZA++ alignments in both directions (Koehn et al., 2003), which we then intersect with the predictions of GIZA++ in the relevant translation direction. This initialisation setup gave the best results (we compare other initialisations in §5.2). The two Gibbs samplers were “burned in” for the first 1000 iterations, after which we ran a further 500 iterations selecting every 50th sample. A phrase table was constructed using these 10 sets of multiple alignments after combining each pair of directional alignments using the *grow-diag-final* heuristic. Using multiple samples in this way constitutes Monte Carlo averaging, which provides a better estimate of uncertainty cf. using a single sample.<sup>5</sup>

The alignment used for the baseline results was produced by combining bidirectional GIZA++ alignments using the *grow-diag-final* heuristic. We used the Moses machine translation decoder (Koehn et al., 2007), using the default features and decoding settings. We compared the performance of Moses using the alignment produced by our model and the baseline alignment, evaluating translation quality using BLEU (Papineni et al., 2002) with case-insensitive  $n$ -gram matching with  $n = 4$ . We used minimum error rate training (Och, 2003) to tune the feature weights to maximise the BLEU score on the development set.

## 5.2 IWSLT Corpus

The first experiments are on the IWSLT data set for Chinese-English translation. The training data consists of 44k sentences from the tourism and travel domain. For the development set we use both ASR devset 1 and 2 from IWSLT 2005, and

<sup>4</sup>All GIZA++ alignments used in our experiments were produced by IBM model4.

<sup>5</sup>The effect on translation scores is modest, roughly amounting to +0.2 BLEU versus using a single sample.

System	Dev	IWSLT05
baseline	45.78	49.98
Markov+ $f_s+e$	49.13	51.54
Markov+ $f_s+e+j$	49.68	52.55
Markov+ $f_s+e+j+f_t$	51.32	53.41

Table 2: Impact of adding factors to our Markov model, showing BLEU scores on IWSLT. Key:  $f_s$ –finish  $e$ –emission  $j$ –jump  $f_t$ –fertility.

for the test set we use the IWSLT 2005 test set. The language model is a 3-gram language model trained using the SRILM toolkit (Stolcke, 2002) on the English side of the training data. Because the data set is small, we performed Gibbs sampling on a single processor.

First we check the effect of the model factors *jump* and *fertility*. Both *emission* and *finish* factors are indispensable to the generative translation process, and consequently these two factors are included in all runs. Table 2 shows translation result for various models, including a baseline and our Markov model with different combinations of factors. Note that even the simplest Markov model far outperforms the GIZA++ baseline (+1.5 BLEU) despite the baseline (IBM model 4) including a number of advanced features (e.g., *jump*, *fertility*) that are not present in the basic Markov model. This improvement is a result of the Markov model making use of rich bilingual contextual information coupled with sophisticated backoff, as opposed to GIZA++ which considers much more local events, with nothing larger than word-class bigrams. Our model shows large improvements as the extra factors are included. *Jump* yields an improvement of +1 BLEU by capturing consistent reordering patterns. Adding *fertility* results in a further +1 BLEU point improvement. Like the IBM models, our approach allows each source word to produce any number of target words. This capacity allows for many non-sensical alignments such as dropping many source words, or aligning single source words to several target words. Explicitly modelling fertility allows for more consistent alignments, especially for special words such as punctuation which usually have a fertility of one.

Next we check the stability of our model with different initialisations. We compare different combination techniques for merging the GIZA++ alignments: *grow-diag-final* (denoted as *gdf*), *intersection* and *grow*. Table 3 shows that the different initialisations have only a small effect on

system	gdf	intersection	grow
baseline	49.98	48.44	50.11
our model	52.96	52.79	53.41

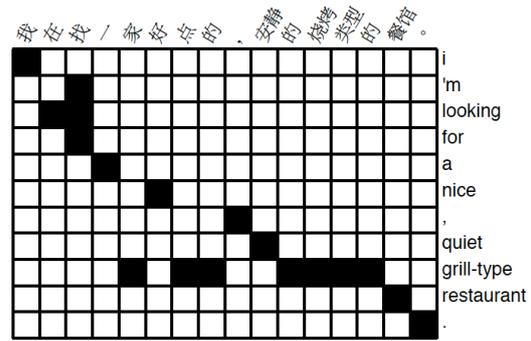
Table 3: Machine translation performance in BLEU % on the IWSLT 2005 Chinese-English test set. The Gibbs samplers were initialized with three different alignments, shown as columns.

the results of our model. While the baseline results vary by up to 1.7 BLEU points for the different alignments, our Markov model provided more stable results with the biggest difference of 0.6. Among the three initialisations, we get the best result with the initialisation of *grow*. *Gdf* often introduces alignment links involving function words which should instead be aligned to null. *Intersection* includes many fewer alignments, typically only between content words, and the sparsity means that words can only have a fertility of either 0 or 1. This leads to the initialisation being a strong mode which is difficult to escape from during sampling. Despite this problem, it has only a mild negative effect on the performance of our model, which is probably due to improvements in the alignments for words that truly should be dropped or aligned only to one word. *Grow* provides a good compromise between *gdf* and *intersection*, and we use this initialisation in all our subsequent experiments.

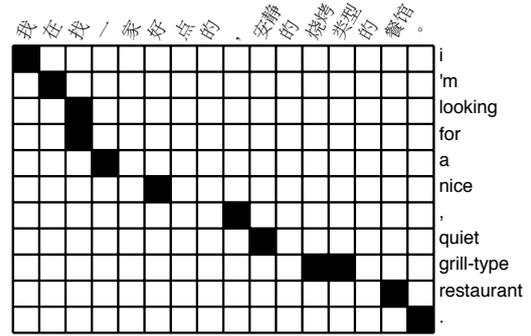
Figure 3 shows an example comparing alignments produced by our model and the GIZA++ baseline, in both cases after combining the two directional models. Note that GIZA++ has linked many function words which should be left unaligned, by using rare English terms as garbage collectors. Consequently this only allows for the extraction of few large phrase-pairs (e.g. <在找, 'm looking for>) and prevents the extraction of some good phrases (e.g. <烧烤类型的, grill-type>, for “家” and “点的” are wrongly aligned to “grill-type”). In contrast, our model better aligns the function words, such that many more useful phrase pairs can be extracted, i.e., <在, 'm>, <找, looking for>, <烧烤类型, grill-type> and their combinations with neighbouring phrase pairs.

### 5.3 FBIS Corpus

Theoretically, Bayesian models should outperform maximum likelihood approaches on small data sets, due to their improved modelling of un-



(a) GIZA++ baseline



(b) our model

Figure 3: Comparison of an alignment inferred by the baseline vs. our approach.

certainty. For larger datasets, however, the difference between the two techniques should narrow. Hence one might expect that upon moving to larger translation datasets our gains might evaporate. This chain of reasoning ignores the fact that our model is considerably richer than the baseline IBM models, in that we model rich contextual correlations between translation decisions, and consequently our approach has a lower inductive bias. For this reason our model should continue to improve with more data, by inferring better estimates of translation decision  $n$ -grams. A caveat though is that inference by sampling becomes less efficient on larger data sets due to stronger modes, requiring more iterations for convergence.

To test whether our improvements carry over to larger datasets, we assess the performance of our model on the FBIS Chinese-English data set. Here the training data consists of the non-UN portions and non-HK Hansards portions of the NIST training corpora distributed by the LDC, totalling 303k sentence pairs with 8m and 9.4m words of Chinese and English, respectively. For the development set we use the NIST 2002 test set, and evaluate performance on the test sets from NIST 2003

	NIST02	NIST03	NIST05
baseline	33.31	30.09	29.01
our model	33.83	31.02	30.23

Table 4: Translation performance on Chinese to English translation, showing BLEU% for models trained on the FBIS data set.

and 2005. The language model is a 3-gram LM trained on Xinhua portion of the Gigaword corpus using the SRILM toolkit with modified Kneser-Ney smoothing. As the FBIS data set is large, we employed 3-processor MPI for each Gibbs sampler, which ran in half the time compared to using a single processor.

Table 4 shows the results on the FBIS data set. Our model outperforms the baseline on both test sets by about 1 BLEU. This provides evidence that our model performs well in the large data setting, with our rich modelling of context still proving useful. The non-parametric nature of the model allows for rich dynamic backoff behaviour such that it can learn accurate models in both high and low data scenarios.

#### 5.4 Arabic English translation

Translation between Chinese and English is very difficult, particularly due to word order differences which are not handled well by phrase-based approaches. In contrast Arabic to English translation needs less reordering, and phrase-based models produce better translations. This translation task is a good test for the generality of our approach. Our Ar-En training data comprises several LDC corpora,<sup>6</sup> using the same experimental setup as in Blunsom et al. (2009a). Overall there are 276k sentence pairs and 8.21m and 8.97m words in Arabic and English, respectively. We evaluate on the NIST test sets from 2003 and 2005, and the 2002 test set was used for MERT training.

Table 5 shows the results. On all test sets our approach outperforms the baseline, and for the NIST03 test set the improvement is substantial, with a +0.74 BLEU improvement. In general the improvements are more modest than for the Chinese-English results above. We suggest that this is due to the structure of Arabic-English translation better suiting the modelling assumptions behind IBM model 4, particularly its bias towards monotone translations. Consequently the addi-

<sup>6</sup>LDC2004E72, LDC2004T17, LDC2004T18, LDC2006T02

	F1%	NIST02	NIST03	NIST05
baseline	64.9	57.00	48.75	48.93
our model	65.7	57.14	49.49	48.96

Table 5: Translation performance on Arabic to English translation, showing BLEU%. Also shown is word-alignment accuracy.

tional context provided by our model is less important. Table 5 also reports alignment results on manually aligned Ar-En sentence pairs,<sup>7</sup> measuring the F1 score for the GIZA++ baseline alignments and the alignment from the final sample with our model.<sup>8</sup> Our model outperforms the baseline, although the improvement is modest.

## 6 Conclusions and Future Work

This paper proposes a word-based Markov model of translation which correlates translation decisions by conditioning on recent decisions, and incorporates a hierarchical Pitman-Yor process prior permitting elaborate backoff behaviour. The model can learn sequences of translation decisions, akin to phrases in standard phrase-based models, while simultaneously learning word level phenomena. This mechanism generalises the concept of phrases in phrase-based MT, while also capturing richer phenomena such as gapping phrases in the source. Experiments show that our model performs well both on the small and large datasets for two different translation tasks, consistently outperforming a competitive baseline. In this paper the model was only used to infer word alignments; in future work we intend to develop a decoding algorithm for directly translating with the model.

### Acknowledgements

This work was supported by the EPSRC (grant EP/I034750/1).

### References

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009a. A Gibbs sampler for phrasal synchronous grammar induction. In *Proc. of ACL-IJCNLP*, pages 782–790.

<sup>7</sup>LDC2012T16

<sup>8</sup>Directional alignments are intersected using the grow-diag-final heuristic.

- Phil Blunsom, Trevor Cohn, Sharon Goldwater, and Mark Johnson. 2009b. A note on the implementation of hierarchical dirichlet processes. In *Proc. of ACL-IJCNLP*, pages 337–340.
- Peter E. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–331.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL*, pages 263–270.
- Josep Maria Crego, François Yvon, and José B. Mariño. 2011. Ncode: an open source bilingual n-gram SMT toolkit. *Prague Bull. Math. Linguistics*, 96:49–58.
- Nadir Durrani, Helmut Schmid, and Alexander Fraser. 2011. A joint sequence translation model with integrated reordering. In *Proc. of ACL:HLT*, pages 1045–1054.
- Michel Galley and Christopher D. Manning. 2010. Accurate non-hierarchical phrase-based translation. In *Proc. of NAACL*, pages 966–974.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. of ACL*, pages 961–968.
- Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proc. of HLT-NAACL*, pages 317–325.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 127–133.
- Philipp Koehn, Hieu Hoang, Alexandra Birch Mayne, Christopher Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL*.
- Hai-Son Le, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proc. of NAACL*, pages 39–48.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. of COLING-ACL*, pages 609–616, July.
- Frans J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51.
- Frans J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL*, pages 160–167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318.
- Jim Pitman and Marc Yor. 1997. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25(2):855–900.
- Andreas Stolcke. 2002. SRILM: An extensible language modeling toolkit. In *Proc. of ICSLP*.
- Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.
- Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proc. of ACL*, pages 985–992.
- Ashish Vaswani, Haitao Mi, Liang Huang, and David Chiang. 2011. Rule markov models for fast tree-to-string translation. In *Proc. of ACL*, pages 856–864.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proc. of COLING*, pages 836–841.

# Scaling Semi-supervised Naive Bayes with Feature Marginals

Michael R. Lucas and Doug Downey

Northwestern University

2133 Sheridan Road

Evanston, IL 60208

mlucas@u.northwestern.edu

ddowney@eecs.northwestern.edu

## Abstract

Semi-supervised learning (SSL) methods augment standard machine learning (ML) techniques to leverage unlabeled data. SSL techniques are often effective in text classification, where labeled data is scarce but large unlabeled corpora are readily available. However, existing SSL techniques typically require multiple passes over the entirety of the unlabeled data, meaning the techniques are not applicable to large corpora being produced today.

In this paper, we show that improving marginal word frequency estimates using unlabeled data can enable semi-supervised text classification that scales to massive unlabeled data sets. We present a novel learning algorithm, which optimizes a Naive Bayes model to accord with statistics calculated from the unlabeled corpus. In experiments with text topic classification and sentiment analysis, we show that our method is both more scalable and more accurate than SSL techniques from previous work.

## 1 Introduction

Semi-supervised Learning (SSL) is a Machine Learning (ML) approach that utilizes large amounts of unlabeled data, combined with a smaller amount of labeled data, to learn a target function (Zhu, 2006; Chapelle et al., 2006). SSL is motivated by a simple reality: the amount of available machine-readable data is exploding, while human capacity for hand-labeling data for any given ML task remains relatively constant. Experiments in text classification and other domains have demonstrated that by leveraging unlabeled data, SSL techniques improve machine learning performance when human input is limited

(e.g., (Nigam et al., 2000; Mann and McCallum, 2010)).

However, current SSL techniques have scalability limitations. Typically, for each target concept to be learned, a semi-supervised classifier is trained using iterative techniques that execute multiple passes over the unlabeled data (e.g., Expectation-Maximization (Nigam et al., 2000) or Label Propagation (Zhu and Ghahramani, 2002)). This is problematic for text classification over large unlabeled corpora like the Web: new target concepts (new tasks and new topics of interest) arise frequently, and performing even a single pass over a large corpus for each new target concept is intractable.

In this paper, we present a new SSL text classification approach that scales to large corpora. Instead of utilizing unlabeled examples directly for each given target concept, our approach is to precompute a small set of *statistics* over the unlabeled data in advance. Then, for a given target class and labeled data set, we utilize the statistics to improve a classifier.

Specifically, we introduce a method that extends Multinomial Naive Bayes (MNB) to leverage marginal probability statistics  $P(w)$  of each word  $w$ , computed over the unlabeled data. The marginal statistics are used as a constraint to improve the class-conditional probability estimates  $P(w|+)$  and  $P(w|-)$  for the positive and negative classes, which are often noisy when estimated over sparse labeled data sets. We refer to the technique as MNB with Frequency Marginals (MNB-FM).

In experiments with large unlabeled data sets and sparse labeled data, we find that MNB-FM is both faster and more accurate on average than standard SSL methods from previous work, including Label Propagation, MNB with Expectation-Maximization, and the recent Semi-supervised Frequency Estimate (SFE) algorithm (Su et al., 2011). We also analyze how MNB-

FM improves accuracy, and find that surprisingly MNB-FM is especially useful for improving class-conditional probability estimates for words that never occur in the training set.

The paper proceeds as follows. We formally define the task in Section 2. Our algorithm is defined in Section 3. We present experimental results in Section 4, and analysis in Section 5. We discuss related work in Section 6 and conclude in Section 7 with a discussion of future work.

## 2 Problem Definition

We consider a semi-supervised classification task, in which the goal is to produce a mapping from an instance space  $\mathcal{X}$  consisting of  $T$ -tuples of non-negative integer-valued features  $\mathbf{w} = (w_1, \dots, w_T)$ , to a binary output space  $\mathcal{Y} = \{-, +\}$ . In particular, our experiments will focus on the case in which the  $w_i$ 's represent word counts in a given document, in a corpus of vocabulary size  $T$ .

We assume the following inputs:

- A set of zero or more *labeled* documents  $D_L = \{(\mathbf{w}^d, y^d) | d = 1, \dots, n\}$ , drawn i.i.d. from a distribution  $P(\mathbf{w}, y)$  for  $\mathbf{w} \in \mathcal{X}$  and  $y \in \mathcal{Y}$ .
- A large set of *unlabeled* documents  $D_U = \{(\mathbf{w}^d) | d = n+1, \dots, n+u\}$  drawn from the marginal distribution  $P(\mathbf{w}) = \sum_y P(\mathbf{w}, y)$ .

The goal of the task is to output a classifier  $f : \mathcal{X} \rightarrow \mathcal{Y}$  that performs well in predicting the classes of given unlabeled documents. The metrics of evaluation we focus on in our experiments are detailed in Section 4.

Our semi-supervised technique utilizes statistics computed over the labeled corpus, denoted as follows. We use  $N_w^+$  to denote the sum of the occurrences of word  $w$  over all documents in the positive class in the labeled data  $D_L$ . Also, let  $N^+ = \sum_{w \in D_L} N_w^+$  be the sum value of all word counts in the labeled positive documents. The count of the remaining words in the positive documents is represented as  $N_{\neg w}^+ = N^+ - N_w^+$ . The quantities  $N^-$ ,  $N_w^-$ , and  $N_{\neg w}^-$  are defined similarly for the negative class.

## 3 MNB with Feature Marginals

We now introduce our algorithm, which scalably utilizes large unlabeled data stores for classifica-

tion tasks. The technique builds upon the multinomial Naive Bayes model, and is denoted as MNB with Feature Marginals (MNB-FM).

### 3.1 MNB-FM Method

In the text classification setting, each feature value  $w^d$  represents count of observations of word  $w$  in document  $d$ . MNB makes the simplifying assumption that word occurrences are conditionally independent of each other given the class (+ or -) of the example. Formally, let the probability  $P(w|+)$  of the  $w$  in the positive class be denoted as  $\theta_w^+$ . Let  $P(+)$  denote the prior probability that a document is of the positive class, and  $P(-) = 1 - P(+)$  the prior for the negative class. Then MNB represents the class probability of an example as:

$$P(+|d) = \frac{\prod_{w \in d} (\theta_w^+)^{w^d} P(+)}{\prod_{w \in d} (\theta_w^-)^{w^d} P(-) + \prod_{w \in d} (\theta_w^+)^{w^d} P(+)} \quad (1)$$

MNB estimates the parameters  $\theta_w^+$  from the corresponding counts in the training set. The maximum-likelihood estimate of  $\theta_w^+$  is  $N_w^+/N^+$ , and to prevent zero-probability estimates we employ ‘‘add-1’’ smoothing (typical in MNB) to obtain the estimate:

$$\theta_w^+ = \frac{N_w^+ + 1}{N^+ + |T|}.$$

After MNB calculates  $\theta_w^+$  and  $\theta_w^-$  from the training set for each feature in the feature space, it can then classify test examples using Equation 1.

MNB-FM attempts to improve MNB’s estimates of  $\theta_w^+$  and  $\theta_w^-$ , using statistics computed over the unlabeled data. Formally, MNB-FM leverages the equality:

$$P(w) = \theta_w^+ P_t(+)+ \theta_w^- P_t(-) \quad (2)$$

The left-hand-side of Equation 2,  $P(w)$ , represents the probability that a given randomly drawn token from the unlabeled data happens to be the word  $w$ . We write  $P_t(+)$  to denote the probability that a randomly drawn *token* (i.e. a word occurrence) from the corpus comes from the positive class. Note that  $P_t(+)$  can differ from  $P(+)$ , the prior probability that a document is positive, due to variations in document length.  $P_t(-)$  is defined similarly for the negative class. MNB-FM is motivated by the insight that the left-hand-side of

Equation 2 can be estimated in advance, without knowledge of the target class, simply by counting the number of tokens of each word in the unlabeled data.

MNB-FM then uses this improved estimate of  $P(w)$  as a constraint to improve the MNB parameters on the right-hand-side of Equation 2. We note that  $P_t(+)$  and  $P_t(-)$ , even for a small training set, can typically be estimated reliably—every token in the training data serves as an observation of these quantities. However, for large and sparse feature spaces common in settings like text classification, many features occur in only a small fraction of examples—meaning  $\theta_w^+$  and  $\theta_w^-$  must be estimated from only a handful of observations. MNB-FM attempts to improve the noisy estimates  $\theta_w^+$  and  $\theta_w^-$  utilizing the robust estimate for  $P(w)$  computed over unlabeled data.

Specifically, MNB-FM proceeds by assuming the MLEs for  $P(w)$  (computed over unlabeled data),  $P_t(+)$ , and  $P_t(-)$  are correct, and re-estimates  $\theta_w^+$  and  $\theta_w^-$  under the constraint in Equation 2.

First, the maximum likelihood estimates of  $\theta_w^+$  and  $\theta_w^-$  given the training data  $D_L$  are:

$$\begin{aligned} & \arg \max_{\theta_w^+, \theta_w^-} P(D_L | \theta_w^+, \theta_w^-) \\ &= \arg \max_{\theta_w^+, \theta_w^-} \theta_w^{+(N_w^+)} (1 - \theta_w^+)^{(N_{-w}^+)} \\ & \quad \theta_w^{-(N_w^-)} (1 - \theta_w^-)^{(N_{-w}^-)} \\ &= \arg \max_{\theta_w^+, \theta_w^-} N_w^+ \ln(\theta_w^+) + N_{-w}^+ \ln(1 - \theta_w^+) + \\ & \quad N_w^- \ln(\theta_w^-) + N_{-w}^- \ln(1 - \theta_w^-) \end{aligned} \quad (3)$$

We can rewrite the constraint in Equation 2 as:

$$\theta_w^- = K - \theta_w^+ L$$

where for compactness we represent:

$$K = \frac{P(w)}{P_t(-)}; L = \frac{P_t(+)}{P_t(-)}.$$

Substituting the constraint into Equation 3 shows that we wish to choose  $\theta_w^+$  as:

$$\begin{aligned} & \arg \max_{\theta_w^+} N_w^+ \ln(\theta_w^+) + N_{-w}^+ \ln(1 - \theta_w^+) + \\ & \quad N_w^- \ln(K - L\theta_w^+) + N_{-w}^- \ln(1 - K + L\theta_w^+) \end{aligned}$$

The optimal values for  $\theta_w^+$  are thus located at the solutions of:

$$0 = \frac{N_w^+}{\theta_w^+} + \frac{N_{-w}^+}{\theta_w^+ - 1} + \frac{LN_w^-}{L\theta_w^+ - K} + \frac{LN_{-w}^-}{L\theta_w^+ - K + 1}$$

Both  $\theta_w^+$  and  $\theta_w^-$  are constrained to valid probabilities in  $[0,1]$  when  $\theta_w^+ \in [0, \frac{K}{L}]$ . If  $N_{-w}^+$  and  $N_w^-$  have non-zero counts, vertical asymptotes exist at 0 and  $\frac{K}{L}$  and guarantee a solution in this range. Otherwise, a valid solution may not exist. In that case, we default to the add-1 Smoothing estimates used by MNB. Finally, after optimizing the values  $\theta_w^+$  and  $\theta_w^-$  for each word  $w$  as described above, we normalize the estimates to obtain valid conditional probability distributions, i.e. with  $\sum_w \theta_w^+ = \sum_w \theta_w^- = 1$

### 3.2 MNB-FM Example

The following concrete example illustrates how MNB-FM can improve MNB parameters using the statistic  $P(w)$  computed over unlabeled data. The example comes from the Reuters Aptemod text classification task addressed in Section 4, using bag-of-words features for the Earnings class. In one experiment with 10 labeled training examples, we observed 5 positive and 5 negative examples, with the word “resources” occurring three times in the set (once in the positive class, twice in the negative class).

MNB uses add-1 smoothing to estimate the conditional probability of the word “resources” in each class as  $\theta_w^+ = \frac{1+1}{216+33504} = 5.93e-5$ , and  $\theta_w^- = \frac{2+1}{547+33504} = 8.81e-5$ . Thus,  $\frac{\theta_w^+}{\theta_w^-} = 0.673$  implying that “resources” is a negative indicator of the Earnings class. However, this estimate is inaccurate. In fact, over the *full* dataset, the parameter values we observe are  $\theta_w^+ = \frac{93}{168549} = 5.70e-4$  and  $\theta_w^- = \frac{263}{564717} = 4.65e-4$ , with a ratio of  $\frac{\theta_w^+}{\theta_w^-} = 1.223$ . Thus, in actuality, the word “resources” is a mild *positive* indicator of the Earnings class. Yet because MNB estimates its parameters from only the sparse training data, it can be inaccurate.

The optimization in MNB-FM seeks to accord its parameter estimates with the feature frequency, computed from unlabeled data, of  $P(w) = 4.89e-4$ . We see that compared with  $P(w)$ , the  $\theta_w^+$  and  $\theta_w^-$  that MNB estimates from the training data are both too low by almost an order of magnitude. Further, the maximum likelihood estimate for  $\theta_w^-$  (based on an occurrence count of 2 out of 547 observations) is somewhat more reliable than that for  $\theta_w^+$  (1 of 216 observations). As a result,  $\theta_w^+$  is adjusted upward relatively *more* than  $\theta_w^-$  via MNB-FM’s constrained ML estimation. MNB-FM returns  $\theta_w^+ = 6.52e-5$  and  $\theta_w^- = 6.04e-5$ . The ratio

$\frac{\theta_w^+}{\theta_w^-}$  is 1.079, meaning MNB-FM correctly identifies the word “resources” as an indicator of the positive class.

The above example illustrates how MNB-FM can leverage frequency marginal statistics computed over unlabeled data to improve MNB’s conditional probability estimates. We analyze how frequently MNB-FM succeeds in improving MNB’s estimates in practice, and the resulting impact on classification accuracy, below.

## 4 Experiments

In this section, we describe our experiments quantifying the accuracy and scalability of our proposed technique. Across multiple domains, we find that MNB-FM outperforms a variety of approaches from previous work.

### 4.1 Data Sets

We evaluate on two text classification tasks: topic classification, and sentiment detection. In topic classification, the task is to determine whether a test document belongs to a specified topic. We train a classifier separately (i.e., in a binary classification setting) for each topic and measure classification performance for each class individually.

The sentiment detection task is to determine whether a document is written with a positive or negative sentiment. In our case, the goal is to determine if the given text belongs to a positive review of a product.

#### 4.1.1 RCV1

The Reuters RCV1 corpus is a standard large corpus used for topic classification evaluations (Lewis et al., 2004). It includes 804,414 documents with several nested target classes. We consider the 5 largest base classes after punctuation and stopwords were removed. The vocabulary consisted of 288,062 unique words, and the total number of tokens in the data set was 99,702,278. Details of the classes can be found in Table 1.

#### 4.1.2 Reuters Aptemod

While MNB-FM is designed to improve the scalability of SSL to large corpora, some of the comparison methods from previous work were not tractable on the large topic classification data set RCV1. To evaluate these methods, we also experimented with the Reuters Aptemod dataset (Yang and Liu, 1999), consisting of 10,788 documents belonging to 90 classes. We consider the 10 most

Class	# Positive
CCAT	381327 (47.40%)
GCAT	239267 (29.74%)
MCAT	204820 (25.46%)
ECAT	119920 (14.91%)
GPOL	56878 (7.07%)

Table 1: RCV1 dataset details

Class	# Positive
Earnings	3964 (36.7%)
Acquisitions	2369 (22.0%)
Foreign	717 (6.6%)
Grain	582 (5.4%)
Crude	578 (5.4%)
Trade	485 (4.5%)
Interest	478 (4.4%)
Shipping	286 (2.7%)
Wheat	283 (2.6%)
Corn	237 (2.2%)

Table 2: Aptemod dataset details

frequent classes, with varying degrees of positive/negative skew. Punctuation and stopwords were removed during preprocessing. The Aptemod data set contained 33,504 unique words and a total of 733,266 word tokens. Details of the classes can be found in Table 2.

#### 4.1.3 Sentiment Classification Data

In the domain of Sentiment Classification, we tested on the Amazon dataset from (Blitzer et al., 2007). Stopwords listed in an included file were ignored for our experiments and we only considered unigram features. Unlike the two Reuters data sets, each category had a unique set of documents of varying size. For our experiments, we only used the 10 largest categories. Details of the categories can be found in Table 3.

In the Amazon Sentiment Classification data set, the task is to determine whether a review is positive or negative based solely on the reviewer’s submitted text. As such, the positive and negative

Class	# Instances	# Positive	Vocabulary
Music	124362	113997 (91.67%)	419936
Books	54337	47767 (87.91%)	220275
Dvd	46088	39563 (85.84%)	217744
Electronics	20393	15918 (78.06%)	65535
Kitchen	18466	14595 (79.04%)	47180
Video	17389	15017 (86.36%)	106467
Toys	12636	10151 (80.33%)	37939
Apparel	8940	7642 (85.48%)	22326
Health	6507	5124 (78.75%)	24380
Sports	5358	4352 (81.22%)	24237

Table 3: Amazon dataset details

labels are equally relevant. For our metrics, we calculate the scores for both the positive and negative class and report the average of the two (in contrast to the Reuters data sets, in which we only report the scores for the positive class).

## 4.2 Comparison Methods

In addition to Multinomial Naive Bayes (discussed in Section 3), we evaluate against a variety of supervised and semi-supervised techniques from previous work, which provide a representation of the state of the art. Below, we detail the comparison methods that we re-implemented for our experiments.

### 4.2.1 NB + EM

We implemented a semi-supervised version of Naive Bayes with Expectation Maximization, based on (Nigam et al., 2000). We found that 15 iterations of EM was sufficient to ensure approximate convergence of the parameters.

We also experimented with different weighting factors to assign to the unlabeled data. While performing per-data-split cross-validation was computationally prohibitive for NB+EM, we performed experiments on one class from each data set that revealed weighting unlabeled examples at 1/5 the weight of a labeled example performed best. We found that our re-implementation of NB+EM slightly outperformed published results on a separate data set (Mann and McCallum, 2010), validating our design choices.

### 4.2.2 Logistic Regression

We implemented Logistic Regression using L2-Normalization, finding this to outperform L1-Normalized and non-normalized versions. The strength of the normalization was selected for each training data set of each size utilized in our experiments.

The strength of the normalization in the logistic regression required cross-validation, which we limited to 20 values logarithmically spaced between  $10^{-4}$  and  $10^4$ . The optimal value was selected based upon the best average F1 score over the 10 folds. We selected a normalization parameter separately for each subset of the training data during experimentation.

### 4.2.3 Label Propagation

For our large unlabeled data set sizes, we found that a standard Label Propagation (LP) approach,

which considers propagating information between all pairs of unlabeled examples, was not tractable. We instead implemented a constrained version of LP for comparison.

In our implementation, we limit the number of edges in the propagation graph. Each node propagates to only to its 10 nearest neighbors, where distance is calculated as the cosine distance between the tf-idf representation of two documents. We found the tf-idf weighting to improve performance over that of simple cosine distance. Propagation was run for 100 iterations or until the entropy dropped below a predetermined threshold, whichever occurred first. Even with these aggressive constraints, Label Propagation was intractable to execute on some of the larger data sets, so we do not report LP results for the RCV1 dataset or for the 5 largest Amazon categories.

### 4.2.4 SFE

We also re-implemented a version of the recent Semi-supervised Frequency Estimate approach (Su et al., 2011). SFE was found to outperform MNB and NB+EM in previous work. Consistent with our MNB implementation, we use Add-1 Smoothing in our SFE calculations although its use is not specifically mentioned in (Su et al., 2011).

SFE also augments multinomial Naive Bayes with the frequency information  $P(w)$ , although in a manner distinct from MNB-FM. In particular, SFE uses the equality  $P(+|w) = P(+, w)/P(w)$  and estimates the rhs using  $P(w)$  computed over all the unlabeled data, rather than using only labeled data as in standard MNB. The primary distinction between MNB-FM and SFE is that SFE adjusts sparse estimates  $P(+, w)$  in the same way as non-sparse estimates, whereas MNB-FM is designed to adjust sparse estimates more than non-sparse ones. Further, it can be shown that as  $P(w)$  of a word  $w$  in the unlabeled data becomes larger than that in the labeled data, SFE's estimate of the ratio  $P(w|+)/P(w|-)$  approaches one. Depending on the labeled data, such an estimate can be arbitrarily inaccurate. MNB-FM does not have this limitation.

## 4.3 Results

For each data set, we evaluate on 50 randomly drawn training splits, each comprised of 1,000 randomly selected documents. Each set included at least one positive and one negative document. We

Data Set	MNB-FM	SFE	MNB	NBEM	LProp	Logist.
Apte (10)	0.306	0.271	<b>0.336</b>	0.306	0.245	0.208
Apte (100)	<b>0.554</b>	0.389	0.222	0.203	0.263	0.330
Apte (1k)	<b>0.729</b>	0.614	0.452	0.321	0.267	0.702
Amzn (10)	<b>0.542</b>	0.524	0.508	0.475	0.470*	0.499
Amzn (100)	<b>0.587</b>	0.559	0.456	0.456	0.498*	0.542
Amzn (1k)	0.687	0.611	0.465	0.455	0.539*	<b>0.713</b>
RCV1 (10)	<b>0.494</b>	0.477	0.387	0.485	-	0.272
RCV1 (100)	<b>0.677</b>	0.613	0.337	0.470	-	0.518
RCV1 (1k)	0.772	0.735	0.408	0.491	-	<b>0.774</b>

\* Limited to 5 of 10 Amazon categories

Table 4: F1, training size in parentheses

respected the order of the training splits such that each sample was a strict subset of any larger training sample of the same split.

We evaluate on the standard metric of F1 with respect to the target class. For Amazon, in which both the “positive” and “negative” classes are potential target classes, we evaluate using macro-averaged scores.

The primary results of our experiments are shown in Table 4. The results show that MNB-FM improves upon the MNB classifier substantially, and also tends to outperform the other SSL and supervised learning methods we evaluated. MNB-FM is the best performing method over all data sets when the labeled data is limited to 10 and 100 documents, except for training sets of size 10 in Aptemod, where MNB has a slight edge.

Tables 5 and 6 present detailed results of the experiments on the RCV1 data set. These experiments are limited to the 5 largest base classes and show the F1 performance of MNB-FM and the various comparison methods, excluding Label Propagation which was intractable on this data set.

Class	MNB-FM	SFE	MNB	NBEM	Logist.
CCAT	0.641	<b>0.643</b>	0.580	0.639	0.532
GCAT	0.639	0.686	0.531	<b>0.732</b>	0.466
MCAT	<b>0.572</b>	0.505	0.393	0.504	0.225
ECAT	<b>0.306</b>	0.267	0.198	0.224	0.096
GPOL	0.313	0.283	0.233	<b>0.326</b>	0.043
Average	<b>0.494</b>	0.477	0.387	0.485	0.272

Table 5: RCV1: F1,  $|D_L|=10$

Class	MNB-FM	SFE	MNB	NBEM	Logist.
CCAT	<b>0.797</b>	0.793	0.624	0.713	0.754
GCAT	<b>0.849</b>	0.848	0.731	0.837	0.831
MCAT	<b>0.776</b>	0.737	0.313	0.516	0.689
ECAT	<b>0.463</b>	0.317	0.017	0.193	0.203
GPOL	<b>0.499</b>	0.370	0.002	0.089	0.114
Average	<b>0.677</b>	0.613	0.337	0.470	0.518

Table 6: RCV1: F1,  $|D_L|=100$

Method	1000	5000	10k	50k	100k
MNB-FM	1.44	1.61	1.69	2.47	5.50
NB+EM	2.95	3.43	4.93	10.07	16.90
MNB	1.15	1.260	1.40	2.20	3.61
Labelprop	0.26	4.17	10.62	67.58	-

Table 7: Runtimes of SSL methods (sec.)

The runtimes of our methods can be seen in Table 7. The results show the runtimes of the SSL methods discussed in this paper as the size of the unlabeled dataset grows. As expected, we find that MNB-FM has runtime similar to MNB, and scales much better than methods that take multiple passes over the unlabeled data.

## 5 Analysis

From our experiments, it is clear that the performance of MNB-FM improves on MNB, and in many cases outperforms all existing SSL algorithms we evaluated. MNB-FM improves the conditional probability estimates in MNB and, surprisingly, we found that it can often improve these estimates for words that do not even occur in the training set.

Tables 8 and 9 show the details of the improvements MNB-FM makes on the feature marginal estimates. We ran MNB-FM and MNB on the RCV1 class MCAT and stored the computed feature marginals for direct comparison. For each word in the vocabulary, we compared each classifier’s conditional probability ratios, i.e.  $\theta^+/\theta^-$ , to the true value over the entire data set. We computed which classifier was closer to the correct ratio for each word. These results were averaged over 5 iterations. From the data, we can see that MNB-FM improves the estimates for many words *not* seen in the training set as well as the most common words, even with small training sets.

### 5.1 Ranking Performance

We also analyzed how well the different methods *rank*, rather than classify, the test documents. We evaluated ranking using the R-precision metric, equal to the precision (i.e. fraction of positive documents classified correctly) of the  $R$  highest-ranked test documents, where  $R$  is the total number of positive test documents.

Logistic Regression performed particularly well on the R-Precision Metric, as can be seen in Tables 10, 11, and 12. Logistic Regression performed less well in the F1 metric. We find that NB+EM

Word Freq.	Fraction Improved vs MNB			Avg Improvement vs MNB			Probability Mass		
	Known	Half Known	Unknown	Known	Half Known	Unknown	Known	Half Known	Unknown
$0-10^{-6}$	-	0.165	<b>0.847</b>	-	-0.805	<b>0.349</b>	-	0.02%	7.69%
$10^{-6}-10^{-5}$	0.200	0.303	<b>0.674</b>	<b>0.229</b>	-0.539	<b>0.131</b>	0.00%	0.54%	14.77%
$10^{-5}-10^{-4}$	0.322	0.348	<b>0.592</b>	-0.597	-0.424	<b>0.025</b>	0.74%	10.57%	32.42%
$10^{-4}-10^{-3}$	<b>0.533</b>	<b>0.564</b>	0.433	<b>0.014</b>	<b>0.083</b>	-0.155	7.94%	17.93%	7.39%
$> 10^{-3}$	-	-	-	-	-	-	-	-	-

Table 8: Analysis of Feature Marginal Improvement of MNB-FM over MNB ( $|D_L| = 10$ ). “Known” indicates words occurring in both positive and negative training examples, “Half Known” indicates words occurring in only positive or negative training examples, while “Unknown” indicates words that never occur in labelled examples. Data is for the RCV1 MCAT category. MNB-FM improves estimates by a substantial amount for unknown words and also the most common known and half-known words.

Word Freq.	Fraction Improved vs MNB			Avg Improvement vs MNB			Probability Mass		
	Known	Half Known	Unknown	Known	Half Known	Unknown	Known	Half Known	Unknown
$0-10^{-6}$	<b>0.567</b>	0.243	<b>0.853</b>	<b>0.085</b>	-0.347	<b>0.143</b>	0.00%	0.22%	7.49%
$10^{-6}-10^{-5}$	0.375	0.310	<b>0.719</b>	-0.213	-0.260	<b>0.087</b>	0.38%	4.43%	10.50%
$10^{-5}-10^{-4}$	0.493	0.426	<b>0.672</b>	-0.071	-0.139	<b>0.067</b>	18.68%	20.37%	4.67%
$10^{-4}-10^{-3}$	<b>0.728</b>	<b>0.669</b>	-	<b>0.233</b>	<b>0.018</b>	-	31.70%	1.56%	-
$> 10^{-3}$	-	-	-	-	-	-	-	-	-

Table 9: Analysis of Feature Marginal Improvement of MNB-FM over MNB ( $|D_L| = 100$ ). Data is for the RCV1 MCAT category (see Table 8). MNB-FM improves estimates by a substantial amount for unknown words and also the most common known and half-known words.

performs particularly well on the R-precision metric on ApteMod, suggesting that its modelling assumptions are more accurate for that particular data set (NB+EM performs significantly worse on the other data sets, however). MNB-FM performs essentially equivalently well, on average, to the best competing method (Logistic Regression) on the large RCV1 data set. However, these experiments show that MNB-FM offers more advantages in document classification than in document ranking.

The ranking results show that LR may be preferred when ranking is important. However, LR underperforms in classification tasks (in terms of F1, Tables 4-6). The reason for this is that LR’s learned classification threshold becomes less accurate when datasets are small and classes are highly

Class	MNB-FM	SFE	MNB	NBEM	LProp	Logist.
Apte (10)	0.353	0.304	0.359	<b>0.631</b>	0.490	0.416
Apte (100)	0.555	0.421	0.343	<b>0.881</b>	0.630	0.609
Apte (1k)	0.723	0.652	0.532	<b>0.829</b>	0.754	0.795
Amzn (10)	0.536	0.527	0.516	0.481	0.535*	<b>0.544</b>
Amzn (100)	0.614	0.562	0.517	0.480	0.573*	<b>0.639</b>
Amzn (1k)	0.717	0.650	0.562	0.483	0.639*	<b>0.757</b>
RCV1 (10)	0.505	0.480	0.421	0.450	-	<b>0.512</b>
RCV1 (100)	0.683	0.614	0.474	0.422	-	<b>0.689</b>
RCV1 (1k)	0.781	0.748	0.535	0.454	-	<b>0.802</b>

\* Limited to 5 of 10 Amazon categories

Table 10: R-Precision, training size in parentheses

skewed. In these cases, LR classifies too frequently in favor of the larger class which is detrimental to its performance. This effect is visible in Tables 5 and 6, where LR’s performance significantly drops for the ECAT and GPOL classes. ECAT and GPOL represent only 14.91% and 7.07% of the RCV1 dataset, respectively.

## 6 Related Work

To our knowledge, MNB-FM is the first approach that utilizes a small set of statistics computed over

Data Set	MNB-FM	SFE	MNB	NBEM	Logist.
CCAT	0.637	0.631	0.620	0.498	<b>0.653</b>
GCAT	0.663	0.711	0.600	<b>0.792</b>	0.671
MCAT	0.580	0.492	0.477	0.510	<b>0.596</b>
ECAT	0.291	0.217	0.214	0.111	<b>0.297</b>
GPOL	<b>0.354</b>	0.352	0.193	0.341	0.341
Average	0.505	0.480	0.421	0.450	<b>0.512</b>

Table 11: RCV1: R-Precision,  $D_L = 10$

Class	MNB-FM	SFE	MNB	NBEM	Logist.
CCAT	0.805	0.797	0.765	0.533	<b>0.809</b>
GCAT	0.849	0.858	0.780	<b>0.869</b>	0.843
MCAT	<b>0.782</b>	0.753	0.579	0.533	0.774
ECAT	0.471	0.293	0.203	0.119	<b>0.498</b>
GPOL	0.509	0.370	0.042	0.056	<b>0.520</b>
Average	0.683	0.614	0.474	0.422	<b>0.689</b>

Table 12: RCV1: R-Precision,  $D_L = 100$

a large unlabeled data set as constraints to improve a semi-supervised classifier. Our experiments demonstrate that MNB-FM outperforms previous approaches across multiple text classification techniques including topic classification and sentiment analysis. Further, the MNB-FM approach offers scalability advantages over most existing semi-supervised approaches.

Current popular Semi-Supervised Learning approaches include using Expectation-Maximization on probabilistic models (e.g. (Nigam et al., 2000)); Transductive Support Vector Machines (Joachims, 1999); and graph-based methods such as Label Propagation (LP) (Zhu and Ghahramani, 2002) and their more recent, more scalable variants (e.g. identifying a small number of representative unlabeled examples (Liu et al., 2010)). In general, these techniques require passes over the entirety of the unlabeled data for each new learning task, intractable for massive unlabeled data sets. Naive implementations of LP cannot scale to large unlabeled data sets, as they have time complexity that increases quadratically with the number of unlabeled examples. Recent LP techniques have achieved greater scalability through the use of parallel processing and heuristics such as Approximate-Nearest Neighbor (Subramanya and Bilmes, 2009), or by decomposing the similarity matrix (Lin and Cohen, 2011). Our approach, by contrast, is to pre-compute a small set of marginal statistics over the unlabeled data, which eliminates the need to scan unlabeled data for each new task. Instead, the complexity of MNB-FM is proportional only to the number of unique words in the labeled data set.

In recent work, Su et al. propose the Semi-supervised Frequency Estimate (SFE), which like MNB-FM utilizes the marginal probabilities of features computed from unlabeled data to improve the Multinomial Naive Bayes (MNB) classifier (Su et al., 2011). SFE has the same scalability advantages as MNB-FM. However, unlike our approach, SFE does not compute maximum-likelihood estimates using the marginal statistics as a constraint. Our experiments show that MNB-FM substantially outperforms SFE.

A distinct method for pre-processing unlabeled data in order to help scale semi-supervised learning techniques involves dimensionality reduction or manifold learning (Belkin and Niyogi, 2004), and for NLP tasks, identifying word representa-

tions from unlabeled data (Turian et al., 2010). In contrast to these approaches, MNB-FM preserves the original feature set and is more scalable (the marginal statistics can be computed in a single pass over the unlabeled data set).

## 7 Conclusion

We presented a novel algorithm for efficiently leveraging large unlabeled data sets for semi-supervised learning. Our MNB-FM technique optimizes a Multinomial Naive Bayes model to accord with statistics of the unlabeled corpus. In experiments across topic classification and sentiment analysis, MNB-FM was found to be more accurate and more scalable than several supervised and semi-supervised baselines from previous work.

In future work, we plan to explore utilizing richer statistics from the unlabeled data, beyond word marginals. Further, we plan to experiment with techniques for unlabeled data sets that also include continuous-valued features. Lastly, we also wish to explore ensemble approaches that combine the best supervised classifiers with the improved class-conditional estimates provided by MNB-FM.

## 8 Acknowledgements

This work was supported in part by DARPA contract D11AP00268.

## References

- Mikhail Belkin and Partha Niyogi. 2004. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56(1):209–239.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics*, Prague, Czech Republic.
- O. Chapelle, B. Schölkopf, and A. Zien, editors. 2006. *Semi-Supervised Learning*. MIT Press, Cambridge, MA.
- Thorsten Joachims. 1999. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397.

- Frank Lin and William W Cohen. 2011. Adaptation of graph-based semi-supervised methods to large-scale text data. In *The 9th Workshop on Mining and Learning with Graphs*.
- Wei Liu, Junfeng He, and Shih-Fu Chang. 2010. Large graph construction for scalable semi-supervised learning. In *ICML*, pages 679–686.
- Gideon S. Mann and Andrew McCallum. 2010. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *J. Mach. Learn. Res.*, 11:955–984, March.
- Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using em. *Mach. Learn.*, 39(2-3):103–134, May.
- Jiang Su, Jelber Sayyad Shirab, and Stan Matwin. 2011. Large scale text classification using semisupervised multinomial naive bayes. In Lise Getoor and Tobias Scheffer, editors, *ICML*, pages 97–104. Omnipress.
- Amar Subramanya and Jeff A. Bilmes. 2009. Entropic graph regularization in non-parametric semi-supervised classification. In *Neural Information Processing Society (NIPS)*, Vancouver, Canada, December.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. *Urbana*, 51:61801.
- Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49. ACM.
- X. Zhu and Z. Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University.
- Xiaojin Zhu. 2006. Semi-supervised learning literature survey.

# Learning Latent Personas of Film Characters

David Bamman   Brendan O'Connor   Noah A. Smith

School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA

{dbamman, brenocon, nasmith}@cs.cmu.edu

## Abstract

We present two latent variable models for learning character types, or *personas*, in film, in which a persona is defined as a set of mixtures over latent lexical classes. These lexical classes capture the stereotypical actions of which a character is the agent and patient, as well as attributes by which they are described. As the first attempt to solve this problem explicitly, we also present a new dataset for the text-driven analysis of film, along with a benchmark testbed to help drive future work in this area.

## 1 Introduction

Philosophers and dramatists have long argued whether the most important element of narrative is *plot* or *character*. Under a classical Aristotelian perspective, plot is supreme;<sup>1</sup> modern theoretical dramatists and screenwriters disagree.<sup>2</sup>

Without addressing this debate directly, much computational work on narrative has focused on learning the sequence of events by which a story is defined; in this tradition we might situate seminal work on learning procedural scripts (Schank and Abelson, 1977; Regneri et al., 2010), narrative chains (Chambers and Jurafsky, 2008), and plot structure (Finlayson, 2011; Elsner, 2012; McIntyre and Lapata, 2010; Goyal et al., 2010).

We present a complementary perspective that addresses the importance of *character* in defining

<sup>1</sup>“Dramatic action . . . is not with a view to the representation of character: character comes in as subsidiary to the actions . . . The Plot, then, is the first principle, and, as it were, the soul of a tragedy: Character holds the second place.” *Poetics* I.VI (Aristotle, 335 BCE).

<sup>2</sup>“Aristotle was mistaken in his time, and our scholars are mistaken today when they accept his rulings concerning character. Character was a great factor in Aristotle’s time, and no fine play ever was or ever will be written without it” (Egri, 1946, p. 94); “What the reader wants is fascinating, complex characters” (McKee, 1997, 100).

a story. Our testbed is film. Under this perspective, a character’s latent internal nature drives the action we observe. Articulating narrative in this way leads to a natural generative story: we first decide that we’re going to make a particular kind of movie (e.g., a romantic comedy), then decide on a set of character types, or *personas*, we want to see involved (the PROTAGONIST, the LOVE INTEREST, the BEST FRIEND). After picking this set, we fill out each of these roles with specific attributes (*female, 28 years old, klutzy*); with this cast of characters, we then sketch out the set of events by which they interact with the world and with each other (*runs but just misses the train, spills coffee on her boss*) – through which they reveal to the viewer those inherent qualities about themselves. This work is inspired by past approaches that infer typed semantic arguments along with narrative schemas (Chambers and Jurafsky, 2009; Regneri et al., 2011), but seeks a more holistic view of character, one that learns from stereotypical attributes in addition to plot events. This work also naturally draws on earlier work on the unsupervised learning of verbal arguments and semantic roles (Pereira et al., 1993; Grenager and Manning, 2006; Titov and Klementiev, 2012) and unsupervised relation discovery (Yao et al., 2011).

This character-centric perspective leads to two natural questions. First, can we learn what those standard personas are by how individual characters (who instantiate those types) are portrayed? Second, can we learn the set of attributes and actions by which we recognize those common types? How do we, as viewers, recognize a VILLIAN?

At its most extreme, this perspective reduces to learning the grand archetypes of Joseph Campbell (1949) or Carl Jung (1981), such as the HERO or TRICKSTER. We seek, however, a more fine-grained set that includes not only archetypes, but stereotypes as well – characters defined by a fixed set of actions widely known to be representative of

a class. This work offers a data-driven method for answering these questions, presenting two probabilistic generative models for inferring latent character types.

This is the first work that attempts to learn explicit character personas in detail; as such, we present a new dataset for character type induction in film and a benchmark testbed for evaluating future work.<sup>3</sup>

## 2 Data

### 2.1 Text

Our primary source of data comes from 42,306 movie plot summaries extracted from the November 2, 2012 dump of English-language Wikipedia.<sup>4</sup> These summaries, which have a median length of approximately 176 words,<sup>5</sup> contain a concise synopsis of the movie’s events, along with implicit descriptions of the characters (e.g., “rebel leader Princess Leia,” “evil lord Darth Vader”). To extract structure from this data, we use the Stanford CoreNLP library<sup>6</sup> to tag and syntactically parse the text, extract entities, and resolve coreference within the document. With this structured representation, we extract linguistic features for each character, looking at immediate verb governors and attribute syntactic dependencies to all of the entity’s mention headwords, extracted from the typed dependency tuples produced by the parser; we refer to “CCprocessed” syntactic relations described in de Marneffe and Manning (2008):

- **Agent verbs.** Verbs for which the entity is an agent argument (*nsubj* or *agent*).
- **Patient verbs.** Verbs for which the entity is the patient, theme or other argument (*dobj*, *nsubjpass*, *iobj*, or any prepositional argument *prep\_\**).
- **Attributes.** Adjectives and common noun words that relate to the mention as adjectival modifiers, noun-noun compounds, appositives, or copulas (*nsubj* or *appos* governors, or *nsubj*, *appos*, *amod*, *nn* dependents of an entity mention).

<sup>3</sup>All datasets and software for replication can be found at <http://www.ark.cs.cmu.edu/personas>.

<sup>4</sup><http://dumps.wikimedia.org/enwiki/>

<sup>5</sup>More popular movies naturally attract more attention on Wikipedia and hence more detail: the top 1,000 movies by box office revenue have a median length of 715 words.

<sup>6</sup><http://nlp.stanford.edu/software/corenlp.shtml>

These three roles capture three different ways in which character personas are revealed: the actions they take on others, the actions done to them, and the attributes by which they are described. For every character we thus extract a bag of  $(r, w)$  tuples, where  $w$  is the word lemma and  $r$  is one of {agent verb, patient verb, attribute} as identified by the above rules.

### 2.2 Metadata

Our second source of information consists of character and movie metadata drawn from the November 4, 2012 dump of Freebase.<sup>7</sup> At the movie level, this includes data on the language, country, release date and detailed genre (365 non-mutually exclusive categories, including “Epic Western,” “Revenge,” and “Hip Hop Movies”). Many of the characters in movies are also associated with the actors who play them; since many actors also have detailed biographical information, we can ground the characters in what we know of those real people – including their gender and estimated age at the time of the movie’s release (the difference between the release date of the movie and the actor’s date of birth).

Across all 42,306 movies, entities average 3.4 agent events, 2.0 patient events, and 2.1 attributes. For all experiments described below, we restrict our dataset to only those events that are among the 1,000 most frequent overall, and only characters with at least 3 events. 120,345 characters meet this criterion; of these, 33,559 can be matched to Freebase actors with a specified gender, and 29,802 can be matched to actors with a given date of birth. Of all actors in the Freebase data whose age is given, the average age at the time of movie is 37.9 (standard deviation 14.1); of all actors whose gender is known, 66.7% are male.<sup>8</sup> The age distribution is strongly bimodal when conditioning on gender: the average age of a female actress at the time of a movie’s release is 33.0 (s.d. 13.4), while that of a male actor is 40.5 (s.d. 13.7).

## 3 Personas

One way we recognize a character’s latent type is by observing the stereotypical actions they

<sup>7</sup><http://download.freebase.com/datadumps/>

<sup>8</sup>Whether this extreme 2:1 male/female ratio reflects an inherent bias in film or a bias in attention on Freebase (or Wikipedia, on which it draws) is an interesting research question in itself.

perform (e.g., VILLAINS *strangle*), the actions done to them (e.g., VILLAINS are *foiled* and *arrested*) and the words by which they are described (VILLAINS are *evil*). To capture this intuition, we define a *persona* as a set of three typed distributions: one for the words for which the character is the agent, one for which it is the patient, and one for words by which the character is attributively modified. Each distribution ranges over a fixed set of latent word classes, or *topics*. Figure 1 illustrates this definition for a toy example: a ZOMBIE persona may be characterized as being the agent of primarily *eating* and *killing* actions, the patient of *killing* actions, and the object of *dead* attributes. The topic labeled *eat* may include words like *eat*, *drink*, and *devour*.

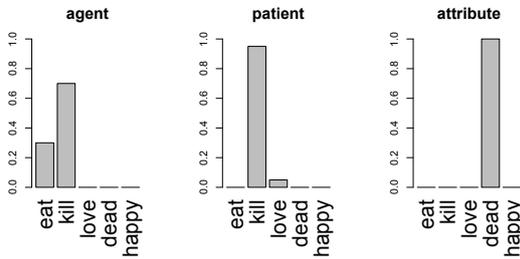
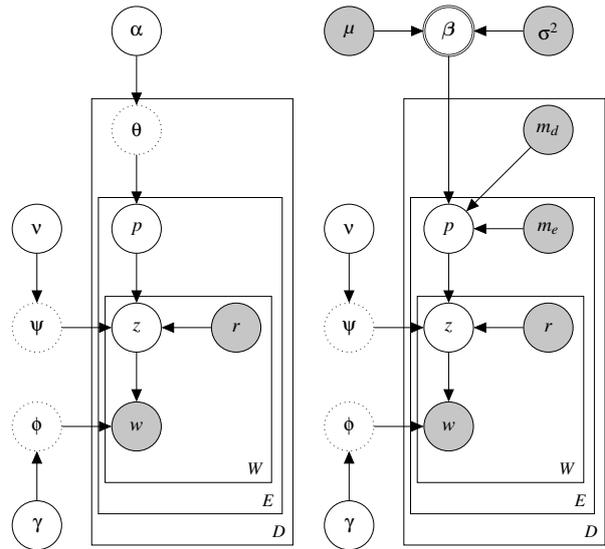


Figure 1: A *persona* is a set of three distributions over latent topics. In this toy example, the ZOMBIE persona is primarily characterized by being the agent of words from the *eat* and *kill* topics, the patient of *kill* words, and the object of words from the *dead* topic.

## 4 Models

Both models that we present here simultaneously learn three things: 1.) a soft clustering over words to topics (e.g., the verb “strangle” is mostly a type of *Assault* word); 2.) a soft clustering over topics to personas (e.g., VILLAINS perform a lot of *Assault* actions); and 3.) a hard clustering over characters to personas (e.g., Darth Vader is a VILLAIN.) They each use different evidence: since our data includes not only textual features (in the form of actions and attributes of the characters) but also non-textual information (such as movie genre, age and gender), we design a model that exploits this additional source of information in discriminating between character types; since this extralinguistic information may not always be available, we also design a model that learns only from the text itself. We present the text-only model first



$P$	Number of personas (hyperparameter)
$K$	Number of word topics (hyperparameter)
$D$	Number of movie plot summaries
$E$	Number of characters in movie $d$
$W$	Number of (role, word) tuples used by character $e$
$\phi_k$	Topic $k$ 's distribution over $V$ words.
$r$	Tuple role: agent verb, patient verb, attribute
$\psi_{p,r}$	Distribution over topics for persona $p$ in role $r$
$\theta_d$	Movie $d$ 's distribution over personas
$p_e$	Character $e$ 's persona (integer, $p \in \{1..P\}$ )
$j$	A specific $(r, w)$ tuple in the data
$z_j$	Word topic for tuple $j$
$w_j$	Word for tuple $j$
$\alpha$	Concentration parameter for Dirichlet model
$\beta$	Feature weights for regression model
$\mu, \sigma^2$	Gaussian mean and variance (for regularizing $\beta$ )
$m_d$	Movie features (from movie metadata)
$m_e$	Entity features (from movie actor metadata)
$\nu_r, \gamma$	Dirichlet concentration parameters

Figure 2: **Above:** Dirichlet persona model (left) and persona regression model (right). **Bottom:** Definition of variables.

for simplicity. Throughout,  $V$  is the word vocabulary size,  $P$  is the number of personas, and  $K$  is the number of topics.

### 4.1 Dirichlet Persona Model

In the most basic model, we only use information from the structured text, which comes as a bag of  $(r, w)$  tuples for each character in a movie, where  $w$  is the word lemma and  $r$  is the relation of the word with respect to the character (one of agent verb, patient verb or attribute, as outlined in §2.1 above). The generative story runs as follows. First, let there be  $K$  latent word topics; as in LDA (Blei et al., 2003), these are words that will be soft-clustered together by virtue of appearing in similar contexts. Each latent word cluster

$\phi_k \sim \text{Dir}(\gamma)$  is a multinomial over the  $V$  words in the vocabulary, drawn from a Dirichlet parameterized by  $\gamma$ . Next, let a persona  $p$  be defined as a set of three multinomials  $\psi_p$  over these  $K$  topics, one for each typed role  $r$ , each drawn from a Dirichlet with a role-specific hyperparameter ( $\nu_r$ ).

Every document (a movie plot summary) contains a set of characters, each of which is associated with a single latent persona  $p$ ; for every observed  $(r, w)$  tuple associated with the character, we sample a latent topic  $k$  from the role-specific  $\psi_{p,r}$ . Conditioned on this topic assignment, the observed word is drawn from  $\phi_k$ . The distribution of these personas for a given document is determined by a document-specific multinomial  $\theta$ , drawn from a Dirichlet parameterized by  $\alpha$ .

Figure 2 (above left) illustrates the form of the model. To simplify inference, we collapse out the persona-topic distributions  $\psi$ , the topic-word distributions  $\phi$  and the persona distribution  $\theta$  for each document. Inference on the remaining latent variables – the persona  $p$  for each character type and the topic  $z$  for each word associated with that character – is conducted via collapsed Gibbs sampling (Griffiths and Steyvers, 2004); at each iteration, for each character  $e$ , we sample their persona  $p_e$ :

$$P(p_e = k \mid p_{-e}, z, \alpha, \nu) \propto \left( c_{d,k}^{-e} + \alpha_k \right) \times \prod_j \frac{(c_{r_j,k,z_j}^{-e} + \nu_{r_j})}{(c_{r_j,k,\star}^{-e} + K\nu_{r_j})} \quad (1)$$

Here,  $c_{d,k}^{-e}$  is the count of all characters in document  $d$  whose current persona sample is also  $k$  (not counting the current character  $e$  under consideration),<sup>9</sup>  $j$  ranges over all  $(r_j, w_j)$  tuples associated with character  $e$ . Each  $c_{r_j,k,z_j}^{-e}$  is the count of all tuples with role  $r_j$  and current topic  $z_j$  used with persona  $k$ .  $c_{r_j,k,\star}^{-e}$  is the same count, summing over all topics  $z$ . In other words, the probability that character  $e$  embodies persona  $k$  is proportional to the number of other characters in the plot summary who also embody that persona (plus the Dirichlet hyperparameter  $\alpha_k$ ) times the contribution of each observed word  $w_j$  for that character, given its current topic assignment  $z_j$ .

Once all personas have been sampled, we sam-

<sup>9</sup>The  $-e$  superscript denotes counts taken without considering the current sample for character  $e$ .

ple the latent topics for each tuple as the following.

$$P(z_j = k \mid p, z_{-j}, w, r, \nu, \gamma) \propto \frac{(c_{r_j,p,k}^{-j} + \nu_{r_j})}{(c_{r_j,p,\star}^{-j} + K\nu_{r_j})} \times \frac{(c_{k,w_j}^{-j} + \gamma)}{(c_{k,\star}^{-j} + V\gamma)} \quad (2)$$

Here, conditioned on the current sample  $p$  for the character’s persona, the probability that tuple  $j$  originates in topic  $k$  is proportional to the number of other tuples with that same role  $r_j$  drawn from the same topic for that persona ( $c_{r_j,p,k}^{-j}$ ), normalized by the number of other  $r_j$  tuples associated with that persona overall ( $c_{r_j,p,\star}^{-j}$ ), multiplied by the number of times word  $w_j$  is associated with that topic ( $c_{k,w_j}^{-j}$ ) normalized by the total number of other words associated with that topic overall ( $c_{k,\star}^{-j}$ ).

We optimize the values of the Dirichlet hyperparameters  $\alpha, \nu$  and  $\gamma$  using slice sampling with a uniform prior every 20 iterations for the first 500 iterations, and every 100 iterations thereafter. After a burn-in phase of 10,000 iterations, we collect samples every 10 iterations (to lessen autocorrelation) until a total of 100 have been collected.

## 4.2 Persona Regression

To incorporate observed metadata in the form of movie genre, character age and character gender, we adopt an “upstream” modeling approach (Mimno and McCallum, 2008), letting those observed features influence the conditional probability with which a given character is expected to assume a particular persona, prior to observing any of their actions. This captures the increased likelihood, for example, that a 25-year-old male actor in an action movie will play an ACTION HERO than he will play a VALLEY GIRL.

To capture these effects, each character’s latent persona is no longer drawn from a document-specific Dirichlet; instead, the  $P$ -dimensional simplex is the output of a multiclass logistic regression, where the document genre metadata  $m_d$  and the character age and gender metadata  $m_e$  together form a feature vector that combines with persona-specific feature weights to form the following log-linear distribution over personas, with the probability for persona  $k$  being:

$$P(p = k \mid m_d, m_e, \beta) = \frac{\exp([m_d; m_e]^\top \beta_k)}{1 + \sum_{j=1}^{P-1} \exp([m_d; m_e]^\top \beta_j)} \quad (3)$$

The persona-specific  $\beta$  coefficients are learned through Monte Carlo Expectation Maximization

(Wei and Tanner, 1990), in which we alternate between the following:

1. Given current values for  $\beta$ , for all characters  $e$  in all plot summaries, sample values of  $p_e$  and  $z_j$  for all associated tuples.
2. Given input metadata features  $m$  and the associated sampled values of  $p$ , find the values of  $\beta$  that maximize the standard multiclass logistic regression log likelihood, subject to  $\ell_2$  regularization.

Figure 2 (above right) illustrates this model. As with the Dirichlet persona model, inference on  $p$  for step 1 is conducted with collapsed Gibbs sampling; the only difference in the sampling probability from equation 1 is the effect of the prior, which here is deterministically fixed as the output of the regression.

$$P(p_e = k \mid p_{-e}, z, \nu, m_d, m_e, \beta) \propto \exp([m_d; m_e]^\top \beta_k) \times \prod_j \frac{(c_{r_j, k, z_j}^- + \nu_{r_j})}{(c_{r_j, k, * }^- + K\nu_{r_j})} \quad (4)$$

The sampling equation for the topic assignments  $z$  is identical to that in equation 2. In practice we optimize  $\beta$  every 1,000 iterations, until a burn-in phase of 10,000 iterations has been reached; at this point we following the same sampling regime as for the Dirichlet persona model.

## 5 Evaluation

We evaluate our methods in two quantitative ways by measuring the degree to which we recover two different sets of gold-standard clusterings. This evaluation also helps offer guidance for model selection (in choosing the number of latent topics and personas) by measuring performance on an objective task.

### 5.1 Character Names

First, we consider all character names that occur in at least two separate movies, generally as a consequence of remakes or sequels; this includes proper names such as “Rocky Balboa,” “Oliver Twist,” and “Indiana Jones,” as well as generic type names such as “Gang Member” and “The Thief”; to minimize ambiguity, we only consider character names consisting of at least two tokens. Each of these names is used by at least two different characters; for example, a character named “Jason Bourne” is portrayed in *The Bourne Identity*, *The Bourne Supremacy*, and *The Bourne Ultimatum*. While

these characters are certainly free to assume different roles in different movies, we believe that, in the aggregate, they should tend to embody the same character type and thus prove to be a natural clustering to recover. 970 character names occur at least twice in our data, and 2,666 individual characters use one of those names. Let those 970 character names define 970 unique gold clusters whose members include the individual characters who use that name.

### 5.2 TV Tropes

As a second external measure of validation, we consider a manually created clustering presented at the website TV Tropes,<sup>10</sup> a wiki that collects user-submitted examples of common tropes (narrative, character and plot devices) found in television, film, and fiction, among other media. While TV Tropes contains a wide range of such conventions, we manually identified a set of 72 tropes that could reasonably be labeled character types, including THE CORRUPT CORPORATE EXECUTIVE, THE HARDBOILED DETECTIVE, THE JERK JOCK, THE KLUTZ and THE SURFER DUDE.

We manually aligned user-submitted examples of characters embodying these 72 character types with the canonical references in Freebase to create a test set of 501 individual characters. While the 72 character tropes represented here are a more subjective measure, we expect to be able to at least partially recover this clustering.

### 5.3 Variation of Information

To measure the similarity between the two clusterings of movie characters, gold clusters  $\mathcal{G}$  and induced latent persona clusters  $\mathcal{C}$ , we calculate the variation of information (Meilă, 2007):

$$VI(\mathcal{G}, \mathcal{C}) = H(\mathcal{G}) + H(\mathcal{C}) - 2I(\mathcal{G}, \mathcal{C}) \quad (5)$$

$$= H(\mathcal{G}|\mathcal{C}) + H(\mathcal{C}|\mathcal{G}) \quad (6)$$

VI measures the information-theoretic distance between the two clusterings: a lower value means greater similarity, and  $VI = 0$  if they are identical. Low VI indicates that (induced) clusters and (gold) clusters tend to overlap; i.e., knowing a character’s (induced) cluster usually tells us their (gold) cluster, and vice versa. Variation of information is a metric (symmetric and obeys triangle

<sup>10</sup><http://tvtropes.org>

		Character Names §5.1			TV Tropes §5.2		
$K$	Model	$P = 25$	$P = 50$	$P = 100$	$P = 25$	$P = 50$	$P = 100$
25	Persona regression	7.73	7.32	6.79	6.26	6.13	5.74
	Dirichlet persona	7.83	7.11	6.44	6.29	6.01	5.57
50	Persona regression	7.59	7.08	6.46	6.30	5.99	5.65
	Dirichlet persona	7.57	7.04	6.35	6.23	5.88	5.60
100	Persona regression	7.58	6.95	6.32	6.11	6.05	5.49
	Dirichlet persona	7.64	6.95	6.25	6.24	5.91	5.42

Table 1: Variation of information between learned personas and gold clusters for different numbers of topics  $K$  and personas  $P$ . Lower values are better. All values are reported in bits.

		Character Names §5.1			TV Tropes §5.2		
$K$	Model	$P = 25$	$P = 50$	$P = 100$	$P = 25$	$P = 50$	$P = 100$
25	Persona regression	62.8 (↑41%)	59.5 (↑40%)	53.7 (↑33%)	42.3 (↑31%)	38.5 (↑24%)	33.1 (↑25%)
	Dirichlet persona	54.7 (↑27%)	50.5 (↑26%)	45.4 (↑17%)	39.5 (↑20%)	31.7 (↑28%)	25.1 (↑21%)
50	Persona regression	63.1 (↑42%)	59.8 (↑42%)	53.6 (↑34%)	42.9 (↑30%)	39.1 (↑33%)	31.3 (↑20%)
	Dirichlet persona	57.2 (↑34%)	49.0 (↑23%)	44.7 (↑16%)	39.7 (↑30%)	31.5 (↑32%)	24.6 (↑22%)
100	Persona regression	63.1 (↑42%)	57.7 (↑39%)	53.0 (↑34%)	43.5 (↑33%)	32.1 (↑28%)	26.5 (↑22%)
	Dirichlet persona	55.3 (↑30%)	49.5 (↑24%)	45.2 (↑18%)	39.7 (↑34%)	29.9 (↑24%)	23.6 (↑19%)

Table 2: Purity scores of recovering gold clusters. Higher values are better. Each absolute purity score is paired with its improvement over a controlled baseline of permuting the learned labels while keeping the cluster proportions the same.

inequality), and has a number of other desirable properties.

Table 1 presents the VI between the learned persona clusters and gold clusters, for varying numbers of personas ( $P = \{25, 50, 100\}$ ) and topics ( $K = \{25, 50, 100\}$ ). To determine significance with respect to a random baseline, we conduct a *permutation test* (Fisher, 1935; Pitman, 1937) in which we randomly shuffle the labels of the learned persona clusters and count the number of times in 1,000 such trials that the VI of the observed persona labels is lower than the VI of the permuted labels; this defines a nonparametric  $p$ -value. All results presented are significant at  $p < 0.001$  (i.e. observed VI is never lower than the simulation VI).

Over all tests in comparison to both gold clusterings, we see VI improve as both  $P$  and, to a lesser extent,  $K$  increase. While this may be expected as the number of personas increase to match the number of distinct types in the gold clusters (970 and 72, respectively), the fact that VI improves as the number of latent topics increases suggests that more fine-grained topics are helpful for capturing nuanced character types.<sup>11</sup>

The difference between the persona regression model and the Dirichlet persona model here is not

<sup>11</sup>This trend is robust to the choice of cluster metric: here VI and  $F$ -score have a correlation of  $-0.87$ ; as more latent topics and personas are added, clustering improves (causing the  $F$ -score to go up and the VI distance to go down).

significant; while VI allows us to compare models with different numbers of latent clusters, its requirement that clusterings be mutually informative places a high overhead on models that are fundamentally unidirectional (in Table 1, for example, the room for improvement between two models of the same  $P$  and  $K$  is naturally smaller than the bigger difference between different  $P$  or  $K$ ). While we would naturally prefer a text-only model to be as expressive as a model that requires potentially hard to acquire metadata, we tease apart whether a distinction actually does exist by evaluating the purity of the gold clusters with respect to the labels assigned them.

## 5.4 Purity

For gold clusters  $\mathcal{G} = \{g_1 \dots g_k\}$  and inferred clusters  $\mathcal{C} = \{c_1 \dots c_j\}$  we calculate purity as:

$$\text{Purity} = \frac{1}{N} \sum_k \max_j |g_k \cap c_j| \quad (7)$$

While purity cannot be used to compare models of different persona size  $P$ , it can help us distinguish between models of the same size. A model can attain perfect purity, however, by placing all characters into a single cluster; to control for this, we present a controlled baseline in which each character is assigned a latent character type label proportional to the size of the latent clusters we have learned (so that, for example, if one latent persona cluster contains 3.2% of the total characters,

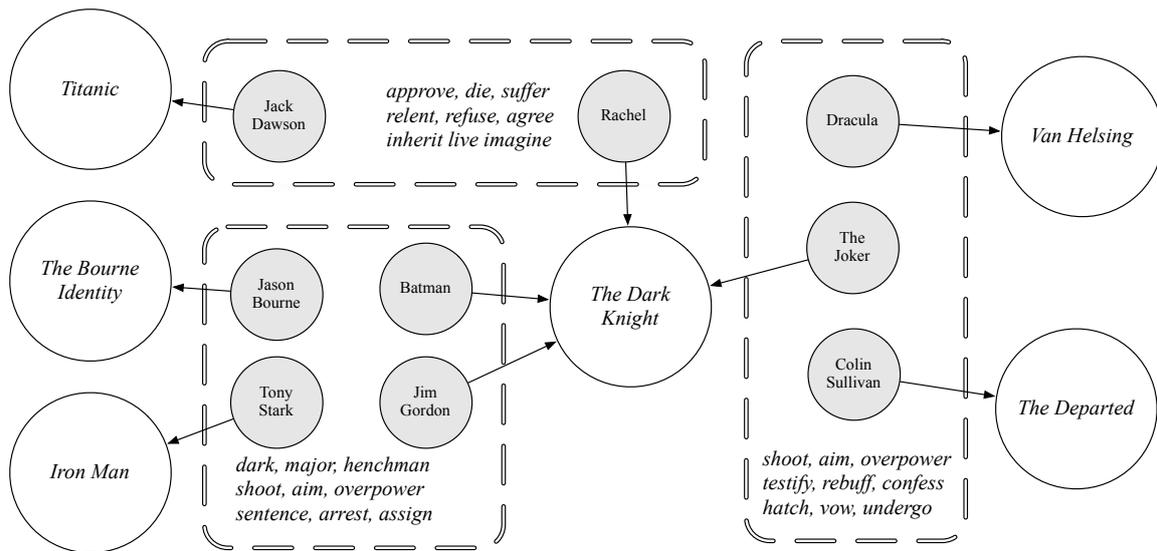


Figure 3: Dramatis personae of *The Dark Knight* (2008), illustrating 3 of the 100 character types learned by the persona regression model, along with links from other characters in those latent classes to other movies. Each character type is listed with the top three latent topics with which it is associated.

the probability of selecting that persona at random is 3.2%). Table 2 presents each model’s absolute purity score paired with its improvement over its controlled permutation (e.g.,  $\uparrow 41\%$ ).

Within each fixed-size partition, the use of metadata yields a substantial improvement over the Dirichlet model, both in terms of absolute purity and in its relative improvement over its sized-controlled baseline. In practice, we find that while the Dirichlet model distinguishes between character personas in different movies, the persona regression model helps distinguish between different personas within the same movie.

## 6 Exploratory Data Analysis

As with other generative approaches, latent persona models enable exploratory data analysis. To illustrate this, we present results from the persona regression model learned above, with 50 latent lexical classes and 100 latent personas. Figure 3 visualizes this data by focusing on a single movie, *The Dark Knight* (2008); the movie’s protagonist, Batman, belongs to the same latent persona as Detective Jim Gordon, as well as other action movie protagonists Jason Bourne and Tony Stark (*Iron Man*). The movie’s antagonist, The Joker, belongs to the same latent persona as Dracula from *Van Helsing* and Colin Sullivan from *The Departed*, illustrating the ability of personas to be informed by, but still cut across, different genres.

Table 3 presents an exhaustive list of all 50 top-

ics, along with an assigned label that consists of the single word with the highest PMI for that class. Of note are topics relating to romance (*unite, marry, woo, elope, court*), commercial transactions (*purchase, sign, sell, owe, buy*), and the classic criminal schema from Chambers (2011) (*sentence, arrest, assign, convict, promote*).

Table 4 presents the most frequent 14 personas in our dataset, illustrated with characters from the 500 highest grossing movies. The personas learned are each three separate mixtures of the 50 latent topics (one for agent relations, one for patient relations, and one for attributes), as illustrated in figure 1 above. Rather than presenting a  $3 \times 50$  histogram for each persona, we illustrate them by listing the most characteristic topics, movie characters, and metadata features associated with it. Characteristic actions and features are defined as those having the highest smoothed pointwise mutual information with that class; exemplary characters are those with the highest posterior probability of being drawn from that class. Among the personas learned are canonical male action heroes (exemplified by the protagonists of *The Bourne Supremacy*, *Speed*, and *Taken*), superheroes (*Hulk*, *Batman and Robin*, *Hector of Troy*) and several romantic comedy types, largely characterized by words drawn from the FLIRT topic, including *flirt, reconcile, date, dance* and *forgive*.

Label	Most characteristic words	Label	Most characteristic words
UNITE	unite marry woo elope court	SWITCH	switch confirm escort report instruct
PURCHASE	purchase sign sell owe buy	INFATUATE	infatuate obsess acquaint revolve concern
SHOOT	shoot aim overpower interrogate kill	ALIEN	alien child governor bandit priest
EXPLORE	explore investigate uncover deduce	CAPTURE	capture corner transport imprison trap
WOMAN	woman friend wife sister husband	MAYA	maya monster monk goon dragon
WITCH	witch villager kid boy mom	INHERIT	inherit live imagine experience share
INVADE	invade sail travel land explore	TESTIFY	testify rebuff confess admit deny
DEFEAT	defeat destroy transform battle inject	APPLY	apply struggle earn graduate develop
CHASE	chase scare hit punch eat	EXPEL	expel inspire humiliate bully grant
TALK	talk tell reassure assure calm	DIG	dig take welcome sink revolve
POP	pop lift crawl laugh shake	COMMAND	command abduct invade seize surrender
SING	sing perform cast produce dance	RELENT	relent refuse agree insist hope
APPROVE	approve die suffer forbid collapse	EMBARK	embark befriend enlist recall meet
WEREWOLF	werewolf mother parent killer father	MANIPULATE	manipulate conclude investigate conduct
DINER	diner grandfather brother terrorist	ELOPE	elope forget succumb pretend like
DECAPITATE	decapitate bite impale strangle stalk	FLEE	flee escape swim hide manage
REPLY	reply say mention answer shout	BABY	baby sheriff vampire knight spirit
DEMON	demon narrator mayor duck crime	BIND	bind select belong refer represent
CONGRATULATE	congratulate cheer thank recommend	REJOIN	rejoin fly recruit include disguise
INTRODUCE	introduce bring mock read hatch	DARK	dark major henchman warrior sergeant
HATCH	hatch don exist vow undergo	SENTENCE	sentence arrest assign convict promote
FLIRT	flirt reconcile date dance forgive	DISTURB	disturb frighten confuse tease scare
ADOPT	adopt raise bear punish feed	RIP	rip vanish crawl drive smash
FAIRY	fairy kidnapper soul slave president	INFILTRATE	infiltrate deduce leap evade obtain
BUG	bug zombie warden king princess	SCREAM	scream faint wake clean hear

Table 3: Latent topics learned for  $K = 50$  and  $P = 100$ . The words shown for each class are those with the highest smoothed PMI, with the label being the single word with the highest PMI.

Freq	Actions	Characters	Features
0.109	DARK <sub>m</sub> , SHOOT <sub>a</sub> , SHOOT <sub>p</sub>	Jason Bourne ( <i>The Bourne Supremacy</i> ), Jack Traven ( <i>Speed</i> ), Jean-Claude ( <i>Taken</i> )	Action, Male, War film
0.079	CAPTURE <sub>p</sub> , INFILTRATE <sub>a</sub> , FLEE <sub>a</sub>	Aang ( <i>The Last Airbender</i> ), Carly ( <i>Transformers: Dark of the Moon</i> ), Susan Murphy/Ginormica ( <i>Monsters vs. Aliens</i> )	Female, Action, Adventure
0.067	DEFEAT <sub>a</sub> , DEFEAT <sub>p</sub> , INFILTRATE <sub>a</sub>	Glenn Talbot ( <i>Hulk</i> ), Batman ( <i>Batman and Robin</i> ), Hector ( <i>Troy</i> )	Action, Animation, Adventure
0.060	COMMAND <sub>a</sub> , DEFEAT <sub>p</sub> , CAPTURE <sub>p</sub>	Zoe Neville ( <i>I Am Legend</i> ), Ursula ( <i>The Little Mermaid</i> ), Joker ( <i>Batman</i> )	Action, Adventure, Male
0.046	INFILTRATE <sub>a</sub> , EXPLORE <sub>a</sub> , EMBARK <sub>a</sub>	Peter Parker ( <i>Spider-Man 3</i> ), Ethan Hunt ( <i>Mission: Impossible</i> ), Jason Bourne ( <i>The Bourne Ultimatum</i> )	Male, Action, Age 34-36
0.036	FLIRT <sub>a</sub> , FLIRT <sub>p</sub> , TESTIFY <sub>a</sub>	Mark Darcy ( <i>Bridget Jones: The Edge of Reason</i> ), Jerry Maguire ( <i>Jerry Maguire</i> ), Donna ( <i>Mamma Mia!</i> )	Female, Romance Film, Comedy
0.033	EMBARK <sub>a</sub> , INFILTRATE <sub>a</sub> , INVADE <sub>a</sub>	Perseus ( <i>Wrath of the Titans</i> ), Maximus Decimus Meridius ( <i>Gladiator</i> ), Julius ( <i>Twins</i> )	Male, Chinese Movies, Spy
0.027	CONGRATULATE <sub>a</sub> , CONGRATULATE <sub>p</sub> , SWITCH <sub>a</sub>	Professor Albus Dumbledore ( <i>Harry Potter and the Philosopher's Stone</i> ), Magic Mirror ( <i>Shrek</i> ), Josephine Anwhistle ( <i>Lemony Snicket's A Series of Unfortunate Events</i> )	Age 58+, Family Film, Age 51-57
0.025	SWITCH <sub>a</sub> , SWITCH <sub>p</sub> , MANIPULATE <sub>a</sub>	Clarice Starling ( <i>The Silence of the Lambs</i> ), Hannibal Lecter ( <i>The Silence of the Lambs</i> ), Colonel Bagley ( <i>The Last Samurai</i> )	Age 58+, Male, Age 45-50
0.022	REPLY <sub>a</sub> , TALK <sub>p</sub> , FLIRT <sub>p</sub>	Graham ( <i>The Holiday</i> ), Abby Richter ( <i>The Ugly Truth</i> ), Anna Scott ( <i>Notting Hill</i> )	Female, Comedy, Romance Film
0.020	EXPLORE <sub>a</sub> , EMBARK <sub>a</sub> , CAPTURE <sub>p</sub>	Harry Potter ( <i>Harry Potter and the Philosopher's Stone</i> ), Harry Potter ( <i>Harry Potter and the Chamber of Secrets</i> ), Captain Leo Davidson ( <i>Planet of the Apes</i> )	Adventure, Family Film, Horror
0.018	FAIRY <sub>m</sub> , COMMAND <sub>a</sub> , CAPTURE <sub>p</sub>	Captain Jack Sparrow ( <i>Pirates of the Caribbean: At World's End</i> ), Shrek ( <i>Shrek</i> ), Shrek ( <i>Shrek Forever After</i> )	Action, Family Film, Animation
0.018	DECAPITATE <sub>a</sub> , DECAPITATE <sub>p</sub> , RIP <sub>a</sub>	Jericho Cane ( <i>End of Days</i> ), Martin Riggs ( <i>Lethal Weapon 2</i> ), Gabriel Van Helsing ( <i>Van Helsing</i> )	Horror, Slasher, Teen
0.017	APPLY <sub>a</sub> , EXPEL <sub>p</sub> , PURCHASE <sub>p</sub>	Oscar ( <i>Shark Tale</i> ), Elizabeth Halsey ( <i>Bad Teacher</i> ), Dre Parker ( <i>The Karate Kid</i> )	Female, Teen, Under Age 22

Table 4: Of 100 latent personas learned, we present the top 14 by frequency. Actions index the latent topic classes presented in table 3; subscripts denote whether the character is predominantly the agent ( $a$ ), patient ( $p$ ) or is modified by an attribute ( $m$ ).

## 7 Conclusion

We present a method for automatically inferring latent character personas from text (and metadata, when available). While our testbed has been textual synopses of film, this approach is easily extended to other genres (such as novelistic fiction) and to non-fictional domains as well, where the choice of portraying a real-life person as embodying a particular kind of persona may, for instance, give insight into questions of media framing and bias in newswire; self-presentation of individual personas likewise has a long history in communication theory (Goffman, 1959) and may be useful for inferring user types for personalization systems (El-Arini et al., 2012). While the goal of this work has been to induce a set of latent character classes and partition all characters among them, one interesting question that remains is how a specific character’s actions may informatively be at odds with their inferred persona, given the choice of that persona as the single best fit to explain the actions we observe. By examining how any individual character deviates from the behavior indicative of their type, we might be able to paint a more nuanced picture of how a character can embody a specific persona while resisting it at the same time.

## Acknowledgments

We thank Megan Morrison at the CMU School of Drama for early conversations guiding our work, as well as the anonymous reviewers for helpful comments. The research reported in this article was supported by U.S. National Science Foundation grant IIS-0915187 and by an ARCS scholarship to D.B. This work was made possible through the use of computing resources made available by the Pittsburgh Supercomputing Center.

## References

- Aristotle. 335 BCE. *Poetics*, translated by Samuel H. Butcher (1902). Macmillan, London.
- David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent dirichlet allocation. *JMLR*, 3:993–1022.
- Joseph Campbell. 1949. *The Hero with a Thousand Faces*. Pantheon Books.
- Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*.
- Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the 47th Annual Meeting of the ACL*.
- Nathanael Chambers. 2011. *Inducing Event Schemas and their Participants from Unlabeled Text*. Ph.D. thesis, Stanford University.
- Marie-Catherine de Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual. Technical report, Stanford University.
- Lajos Egri. 1946. *The Art of Dramatic Writing*. Simon and Schuster, New York.
- Khalid El-Arini, Ulrich Paquet, Ralf Herbrich, Jurgen Van Gael, and Blaise Agüera y Arcas. 2012. Transparent user models for personalization. In *Proceedings of the 18th ACM SIGKDD*.
- Micha Elsner. 2012. Character-based kernels for novelistic plot structure. In *Proceedings of the 13th Conference of the EACL*.
- Mark Alan Finlayson. 2011. *Learning Narrative Structure from Annotated Folktales*. Ph.D. thesis, MIT.
- R. A. Fisher. 1935. *The Design of Experiments*. Oliver and Boyd, Edinburgh and London.
- Erving Goffman. 1959. *The Presentation of the Self in Everyday Life*. Anchor.
- Amit Goyal, Ellen Riloff, and Hal Daumé, III. 2010. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on EMNLP*.
- Trond Grenager and Christopher D. Manning. 2006. Unsupervised discovery of a statistical verb lexicon. In *Proceedings of the 2006 Conference on EMNLP*.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *PNAS*, 101(suppl. 1):5228–5235.
- Carl Jung. 1981. *The Archetypes and The Collective Unconscious*, volume 9 of *Collected Works*. Bollingen, Princeton, NJ, 2nd edition.
- Neil McIntyre and Mirella Lapata. 2010. Plot induction and evolutionary search for story generation. In *Proceedings of the 48th Annual Meeting of the ACL*. Association for Computational Linguistics.
- Robert McKee. 1997. *Story: Substance, Structure, Style and the Principles of Screenwriting*. Harper-Collins.
- Marina Meilă. 2007. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895.
- David Mimno and Andrew McCallum. 2008. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In *Proceedings of UAI*.

- Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of english words. In *Proceedings of the 31st Annual Meeting of the ACL*.
- E. J. G. Pitman. 1937. Significance tests which may be applied to samples from any population. *Supplement to the Journal of the Royal Statistical Society*, 4(1):119–130.
- Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning script knowledge with web experiments. In *Proceedings of the 48th Annual Meeting of the ACL*.
- Michaela Regneri, Alexander Koller, Josef Ruppenhofer, and Manfred Pinkal. 2011. Learning script participants from unlabeled data. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*.
- Roger C. Schank and Robert P. Abelson. 1977. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Lawrence Erlbaum, Hillsdale, NJ.
- Ivan Titov and Alexandre Klementiev. 2012. A bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of EACL*.
- Greg C. G. Wei and Martin A. Tanner. 1990. A Monte Carlo implementation of the EM algorithm and the poor man’s data augmentation algorithms. *Journal of the American Statistical Association*, 85:699–704.
- Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the Conference on EMNLP*.

# Scalable Decipherment for Machine Translation via Hash Sampling

Sujith Ravi

Google

Mountain View, CA 94043

sravi@google.com

## Abstract

In this paper, we propose a new Bayesian inference method to train statistical machine translation systems using only non-parallel corpora. Following a probabilistic *decipherment* approach, we first introduce a new framework for decipherment training that is flexible enough to incorporate any number/type of features (besides simple bag-of-words) as side-information used for estimating translation models. In order to perform fast, efficient Bayesian inference in this framework, we then derive a *hash sampling* strategy that is inspired by the work of Ahmed et al. (2012). The new translation hash sampler enables us to scale elegantly to complex models (for the first time) and large vocabulary/corpora sizes. We show empirical results on the OPUS data—our method yields the best BLEU scores compared to existing approaches, while achieving significant computational speedups (several orders faster). We also report for the first time—BLEU score results for a large-scale MT task using only non-parallel data (EMEA corpus).

## 1 Introduction

Statistical machine translation (SMT) systems these days are built using large amounts of bilingual parallel corpora. The parallel corpora are used to estimate translation model parameters involving word-to-word translation tables, fertilities, distortion, phrase translations, syntactic transformations, etc. But obtaining parallel data is an expensive process and not available for all language

pairs or domains. On the other hand, monolingual data (in written form) exists and is easier to obtain for many languages. Learning translation models from monolingual corpora could help address the challenges faced by modern-day MT systems, especially for low resource language pairs. Recently, this topic has been receiving increasing attention from researchers and new methods have been proposed to train statistical machine translation models using only monolingual data in the source and target language. The underlying motivation behind most of these methods is that statistical properties for linguistic elements are shared across different languages and some of these similarities (mappings) could be automatically identified from large amounts of monolingual data.

The MT literature does cover some prior work on extracting or augmenting partial lexicons using non-parallel corpora (Rapp, 1995; Fung and McKeown, 1997; Koehn and Knight, 2000; Haghghi et al., 2008). However, none of these methods attempt to train end-to-end MT models, instead they focus on mining bilingual lexicons from monolingual corpora and often they require parallel seed lexicons as a starting point. Some of them (Haghghi et al., 2008) also rely on additional linguistic knowledge such as orthography, etc. to mine word translation pairs across related languages (e.g., Spanish/English). Unsupervised training methods have also been proposed in the past for related problems in decipherment (Knight and Yamada, 1999; Snyder et al., 2010; Ravi and Knight, 2011a) where the goal is to decode unknown scripts or ciphers.

The body of work that is more closely related to ours include that of Ravi and Knight (2011b) who introduced a *decipherment* approach for training translation models using only monolingual cor-

pora. Their best performing method uses an EM algorithm to train a word translation model and they show results on a Spanish/English task. Nuhn et al. (2012) extend the former approach and improve training efficiency by pruning translation candidates prior to EM training with the help of context similarities computed from monolingual corpora.

In this work we propose a new Bayesian inference method for estimating translation models from scratch using only monolingual corpora. Secondly, we introduce a new feature-based representation for sampling translation candidates that allows one to incorporate any amount of additional features (beyond simple bag-of-words) as side-information during decipherment training. Finally, we also derive a new accelerated sampling mechanism using locality sensitive hashing inspired by recent work on fast, probabilistic inference for unsupervised clustering (Ahmed et al., 2012). The new sampler allows us to perform fast, efficient inference with more complex translation models (than previously used) and scale better to large vocabulary and corpora sizes compared to existing methods as evidenced by our experimental results on two different corpora.

## 2 Decipherment Model for Machine Translation

We now describe the decipherment problem formulation for machine translation.

**Problem Formulation:** Given a source text  $f$  (i.e., source word sequences  $f_1 \dots f_m$ ) and a monolingual target language corpus, our goal is to decipher the source text and produce a target translation.

Contrary to standard machine translation training scenarios, here we have to estimate the translation model  $P_\theta(f|e)$  parameters using only monolingual data. During decipherment training, our objective is to estimate the model parameters in order to maximize the probability of the source text  $f$  as suggested by Ravi and Knight (2011b).

$$\arg \max_{\theta} \prod_f \sum_e P(e) \cdot P_\theta(f|e) \quad (1)$$

For  $P(e)$ , we use a word n-gram language model (LM) trained on monolingual target text. We then estimate the parameters of the translation model  $P_\theta(f|e)$  during training.

**Translation Model:** Machine translation is a much more complex task than solving other decipherment tasks such as word substitution ciphers (Ravi and Knight, 2011b; Dou and Knight, 2012). The mappings between languages involve non-determinism (i.e., words can have multiple translations), re-ordering of words can occur as grammar and syntax varies with language, and in addition word insertion and deletion operations are also involved.

Ideally, for the translation model  $P(f|e)$  we would like to use well-known statistical models such as IBM Model 3 and estimate its parameters  $\theta$  using the EM algorithm (Dempster et al., 1977). But training becomes intractable with complex translation models and scalability is also an issue when large corpora sizes are involved and the translation tables become huge to fit in memory. So, instead we use a simplified generative process for the translation model as proposed by Ravi and Knight (2011b) and used by others (Nuhn et al., 2012) for this task:

1. Generate a target (e.g., English) string  $e = e_1 \dots e_l$ , with probability  $P(e)$  according to an n-gram language model.
2. Insert a NULL word at any position in the English string, with uniform probability.
3. For each target word token  $e_i$  (including NULLs), choose a source word translation  $f_i$ , with probability  $P_\theta(f_i|e_i)$ . The source word may be NULL.
4. Swap any pair of adjacent source words  $f_{i-1}, f_i$ , with probability  $P(\text{swap})$ ; set to 0.1.
5. Output the foreign string  $f = f_1 \dots f_m$ , skipping over NULLs.

Previous approaches (Ravi and Knight, 2011b; Nuhn et al., 2012) use the EM algorithm to estimate all the parameters  $\theta$  in order to maximize likelihood of the foreign corpus. Instead, we propose a new Bayesian inference framework to estimate the translation model parameters. In spite of using Bayesian inference which is typically slow in practice (with standard Gibbs sampling), we show later that our method is scalable and permits decipherment training using more complex translation models (with several additional parameters).

## 2.1 Adding Phrases, Flexible Reordering and Fertility to Translation Model

We now extend the generative process (described earlier) to more complex translation models.

**Non-local Re-ordering:** The generative process described earlier limits re-ordering to local or adjacent word pairs in a source sentence. We extend this to allow re-ordering between any pair of words in the sentence.

**Fertility:** We also add a fertility model  $P_{\theta_{fert}}$  to the translation model using the formula:

$$P_{\theta_{fert}} = \prod_i n_{\theta}(\phi_i|e_i) \cdot p_1^{\phi_0} \quad (2)$$

$$n_{\theta}(\phi_i|e_i) = \frac{\alpha_{fert} \cdot P_0(\phi_i|e_i) + C^{-i}(e_i, \phi_i)}{\alpha_{fert} + C^{-i}(e_i)} \quad (3)$$

where,  $P_0$  represents the base distribution (which is set to uniform) in a Chinese Restaurant Process (CRP)<sup>1</sup> for the fertility model and  $C^{-i}$  represents the count of events occurring in the history excluding the observation at position  $i$ .  $\phi_i$  is the number of source words aligned to (i.e., generated by) the target word  $e_i$ . We use sparse Dirichlet priors for all the translation model components.<sup>2</sup>  $\phi_0$  represents the target NULL word fertility and  $p_1$  is the insertion probability which is fixed to 0.1. In addition, we set a maximum threshold for fertility values  $\phi_i \leq \gamma \cdot m$ , where  $m$  is the length of the source sentence. This discourages a particular target word (e.g., NULL word) from generating too many source words in the same sentence. In our experiments, we set  $\gamma = 0.3$ . We enforce this constraint in the training process during sampling.<sup>3</sup>

**Modeling Phrases:** Finally, we extend the translation candidate set in  $P_{\theta}(f_i|e_i)$  to model phrases in addition to words for the target side (i.e.,  $e_i$  can now be a word or a phrase<sup>4</sup> previously seen in the monolingual target corpus). This greatly increases the training time since in each sampling step, we now have many more  $e_i$  candidates to choose from. In Section 4, we describe how we deal

<sup>1</sup>Each component in the translation model (word/phrase translations  $P_{\theta}(f_i|e_i)$ , fertility  $P_{\theta_{fert}}$ , etc.) is modeled using a CRP formulation.

<sup>2</sup>i.e., All the concentration parameters are set to low values;  $\alpha_{f|e} = \alpha_{fert} = 0.01$ .

<sup>3</sup>We only apply this constraint when training on source text/corpora made of long sentences (>10 words) where the sampler might converge very slowly. For short sentences, a sparse prior on fertility  $\alpha_{fert}$  typically discourages a target word from being aligned to too many different source words.

<sup>4</sup>Phrase size is limited to two words in our experiments.

with this problem by using a fast, efficient sampler based on hashing that allows us to speed up the Bayesian inference significantly whereas standard Gibbs sampling would be extremely slow.

## 3 Feature-based representation for Source and Target

The model described in the previous section while being flexible in describing the translation process, poses several challenges for training. As the source and target vocabulary sizes increase the size of the translation table ( $|V_f| \cdot |V_e|$ ) increases significantly and often becomes too huge to fit in memory. Additionally, performing Bayesian inference with such a complex model using standard Gibbs sampling can be very slow in practice. Here, we describe a new method for doing Bayesian inference by first introducing a feature-based representation for the source and target words (or phrases) from which we then derive a novel proposal distribution for sampling translation candidates.

We represent both source and target words in a *vector space* similar to how documents are represented in typical information retrieval settings. But unlike documents, here each word  $\mathbf{w}$  is associated with a feature vector  $w^1 \dots w^d$  (where  $w^i$  represents the weight for the feature indexed by  $i$ ) which is constructed from monolingual corpora. For instance, *context features* for word  $\mathbf{w}$  may include other words (or phrases) that appear in the immediate context (n-gram window) surrounding  $\mathbf{w}$  in the monolingual corpus. Similarly, we can add other features based on *topic models*, *orthography* (Haghighi et al., 2008), *temporal* (Klementiev et al., 2012), etc. to our representation all of which can be extracted from monolingual corpora.

Next, given two high dimensional vectors  $\mathbf{u}$  and  $\mathbf{v}$  it is possible to calculate the similarity between the two words denoted by  $s(\mathbf{u}, \mathbf{v})$ . The feature construction process is described in more detail below:

**Target Language:** We represent each word (or phrase)  $e_i$  with the following contextual features along with their counts: (a)  $f_{-context}$ : every (word n-gram, position) pair immediately preceding  $e_i$  in the monolingual corpus (n=1, position=-1), (b) similar features  $f_{+context}$  to model the context following  $e_i$ , and (c) we also throw in generic context features  $f_{scontext}$  without position information—every word that co-occurs with  $e_i$  in the same sen-

tence. While the two position-features provide specific context information (may be sparse for large monolingual corpora), this feature is more generic and captures long-distance co-occurrence statistics.

**Source Language:** Words appearing in a source sentence  $f$  are represented using the corresponding target translation  $e = e_1 \dots e_m$  generated for  $f$  in the current sample during training. For each source word  $f_j \in f$ , we look at the corresponding word  $e_j$  in the target translation. We then extract all the context features of  $e_j$  in the target translation sample sentence  $e$  and add these features ( $f_{-context}, f_{+context}, f_{scontext}$ ) with weights to the feature representation for  $f_j$ .

Unlike the target word feature vectors (which can be pre-computed from the monolingual target corpus), the feature vector for every source word  $f_j$  is dynamically constructed from the target translation sampled in each training iteration. This is a key distinction of our framework compared to previous approaches that use contextual similarity (or any other) features constructed from static monolingual corpora (Rapp, 1995; Koehn and Knight, 2000; Nuhn et al., 2012).

Note that as we add more and more features for a particular word (by training on larger monolingual corpora or adding new types of features, etc.), it results in the feature representation becoming more sparse (especially for source feature vectors) which can cause problems in efficiency as well as robustness when computing similarity against other vectors. In the next section, we will describe how we mitigate this problem by projecting into a low-dimensional space by computing hash signatures.

In all our experiments, we only use the features described above for representing source and target words. We note that the new sampling framework is easily extensible to many additional feature types (for example, monolingual topic model features, etc.) which can be efficiently handled by our inference algorithm and could further improve translation performance but we leave this for future work.

#### 4 Bayesian MT Decipherment via Hash Sampling

The next step is to use the feature representations described earlier and iteratively sample a target word (or phrase) translation candidate  $e_i$  for every

word  $f_i$  in the source text  $f$ . This involves choosing from  $|V_e|$  possible target candidates in every step which can be highly inefficient (and infeasible for large vocabulary sizes). One possible strategy is to compute similarity scores  $s(\mathbf{w}_{f_i}, \mathbf{w}_{e'})$  between the current source word feature vector  $\mathbf{w}_{f_i}$  and feature vectors  $\mathbf{w}_{e' \in V_e}$  for all possible candidates in the target vocabulary. Following this, we can prune the translation candidate set by keeping only the top candidates  $e^*$  according to the similarity scores. Nuhn et al. (2012) use a similar strategy to obtain a more compact translation table that improves runtime efficiency for EM training. Their approach requires calculating and sorting all  $|V_e| \cdot |V_f|$  distances in time  $O(V^2 \cdot \log(V))$ , where  $V = \max(|V_e|, |V_f|)$ .

**Challenges:** Unfortunately, there are several additional challenges which makes inference very hard in our case. Firstly, we would like to include as many features as possible to represent the source/target words in our framework besides simple bag-of-words context similarity (for example, left-context, right-context, and other general-purpose features based on topic models, etc.). This makes the complexity far worse (in practice) since the dimensionality of the feature vectors  $d$  is a much higher value than  $|V_e|$ . Computing similarity scores alone (naïvely) would incur  $O(|V_e| \cdot d)$  time which is prohibitively huge since we have to do this for every token in the source language corpus. Secondly, for Bayesian inference we need to sample from a distribution that involves computing probabilities for all the components (language model, translation model, fertility, etc.) described in Equation 1. This distribution needs to be computed for every source word token  $f_i$  in the corpus, for all possible candidates  $e_i \in V_e$  and the process has to be repeated for multiple sampling iterations (typically more than 1000). Doing standard collapsed Gibbs sampling in this scenario would be very slow and intractable.

We now present an alternative fast, efficient inference strategy that overcomes many of the challenges described above and helps accelerate the sampling process significantly. First, we set our translation models within the context of a more generic and widely known family of distributions—mixtures of exponential families. Then we derive a novel proposal distribution for sampling translation candidates and introduce a new sampler for decipherment training that

is based on locality sensitive hashing (LSH).

Hashing methods such as LSH have been widely used in the past in several scenarios including NLP applications (Ravichandran et al., 2005). Most of these approaches employ LSH within heuristic methods for speeding up nearest-neighbor look up and similarity computation techniques. However, we use LSH hashing within a probabilistic framework which is very different from the typical use of LSH.

Our work is inspired by some recent work by Ahmed et al. (2012) on speeding up Bayesian inference for unsupervised clustering. We use a similar technique as theirs but a different approximate distribution for the proposal, one that is better-suited for machine translation models and without some of the additional overhead required for computing certain terms in the original formulation.

**Mixtures of Exponential Families:** The translation models described earlier (Section 2) can be represented as mixtures of exponential families, specifically mixtures of multinomials. In exponential families, distributions over random variables are given by:

$$p(x; \theta) = \exp(\langle \phi(x), \theta \rangle) - g(\theta) \quad (4)$$

where,  $\phi : \mathcal{X} \rightarrow \mathcal{F}$  is a map from  $x$  to the space of sufficient statistics and  $\theta \in \mathcal{F}$ . The term  $g(\theta)$  ensures that  $p(x; \theta)$  is properly normalized.  $\mathcal{X}$  is the domain of observations  $X = x_1, \dots, x_m$  drawn from some distribution  $p$ . Our goal is to estimate  $p$ . In our case, this refers to the translation model from Equation 1.

We also choose corresponding conjugate Dirichlet distributions for priors which have the property that the posterior distribution  $p(\theta|X)$  over  $\theta$  remains in the same family as  $p(\theta)$ .

Note that the (translation) model in our case consists of multiple exponential families components—a multinomial pertaining to the language model (which remains fixed<sup>5</sup>), and other components pertaining to translation probabilities  $P_\theta(f_i|e_i)$ , fertility  $P_{\theta_{fert}}$ , etc. To do collapsed Gibbs sampling under this model, we would perform the following steps during sampling:

1. For a given source word token  $f_i$  draw target

translation

$$\begin{aligned} e_i &\sim p(e_i|F, E^{-i}) \\ &\propto p(e) \cdot p(f_i|e_i, F^{-i}, E^{-i}) \\ &\quad \cdot p_{fert}(\cdot|e_i, F^{-i}, E^{-i}) \cdot \dots \end{aligned} \quad (5)$$

where,  $F$  is the full source text and  $E$  the full target translation generated during sampling.

2. Update the sufficient statistics for the changed target translation assignments.

For large target vocabularies, computing  $p(f_i|e_i, F^{-i}, E^{-i})$  dominates the inference procedure. We can accelerate this step significantly using a good proposal distribution via hashing.

**Locality Sensitive Hash Sampling:** For general exponential families, here is a Taylor approximation for the data likelihood term (Ahmed et al., 2012):

$$p(x|\cdot) \approx \exp(\langle \phi(x), \theta^* \rangle) - g(\theta^*) \quad (6)$$

where,  $\theta^*$  is the expected parameter (sufficient statistics).

For sampling the translation model, this involves computing an expensive inner product  $\langle \phi(f_i), \theta_{e'}^* \rangle$  for each source word  $f_i$  which has to be repeated for every translation candidate  $e'$ , including candidates that have very low probabilities and are unlikely to be chosen as the translation for  $f_j$ .

So, during decipherment training a standard collapsed Gibbs sampler will waste most of its time on expensive computations that will be discarded in the end anyways. Also, unlike some standard generative models used in other unsupervised learning scenarios (e.g., clustering) that model only observed features (namely words appearing in the document), here we would like to enrich the translation model with a lot more features (side-information).

Instead, we can accelerate the computation of the inner product  $\langle \phi(f_i), \theta_{e'}^* \rangle$  using a *hash sampling* strategy similar to (Ahmed et al., 2012). The underlying idea here is to use binary hashing (Charikar, 2002) to explore only those candidates  $e'$  that are sufficiently close to the best matching translation via a proposal distribution. Next, we briefly introduce some notations and existing theoretical results related to binary hashing before describing the hash sampling procedure.

For any two vectors  $u, v \in \mathbb{R}^n$ ,

$$\langle u, v \rangle = \|u\| \cdot \|v\| \cdot \cos \angle(u, v) \quad (7)$$

<sup>5</sup>A high value for the LM concentration parameter  $\alpha$  ensures that the LM probabilities do not deviate too far from the original fixed base distribution during sampling.

$$\angle(u, v) = \pi Pr\{sgn[\langle u, w \rangle] \neq sgn[\langle v, w \rangle]\} \quad (8)$$

where,  $w$  is a random vector drawn from a symmetric spherical distribution and the term inside  $Pr\{\cdot\}$  represents the relation between the signs of the two inner products.

Let  $h^l(v) \in \{0, 1\}^l$  be an  $l$ -bit binary hash of  $v$  where:  $[h^l(v)]_i := sgn[\langle v, w_i \rangle]$ ;  $w_i \sim U_m$ . Then the probability of matching signs is given by:

$$z^l(u, v) := \frac{1}{l} \|h(u) - h(v)\|_1 \quad (9)$$

So,  $z^l(u, v)$  measures how many bits differ between the hash vectors  $h(u)$  and  $h(v)$  associated with  $u, v$ . Combining this with Equations 6 and 7 we can estimate the unnormalized log-likelihood of a source word  $f_i$  being translated as target  $e'$  via:

$$s^l(f_i, e') \propto \|\theta_{e'}\| \cdot \|\phi(f_i)\| \cdot \cos \pi z^l(\phi(f_i), \theta_{e'}) \quad (10)$$

For each source word  $f_i$ , we now sample from this new distribution (after normalization) instead of the original one. The binary hash representation for the two vectors yield significant speedups during sampling since Hamming distance computation between  $h(u)$  and  $h(v)$  is highly optimized on modern CPUs. Hence, we can compute an estimate for the inner product quite efficiently.<sup>6</sup>

**Updating the hash signatures:** During training, we compute the target candidate projection  $h(\theta_{e'})$  and corresponding norm only once<sup>7</sup> which is different from the setup of Ahmed et al. (2012). The source word projection  $\phi(f_i)$  is dynamically updated in every sampling step. Note that doing this naïvely would scale slowly as  $O(Dl)$  where  $D$  is the total number of features but instead we can update the hash signatures in a more efficient manner that scales as  $O(D_{i>0}l)$  where  $D_{i>0}$  is the number of non-zero entries in the feature representation for the source word  $\phi(f_i)$ . Also, we do not need to store the random vectors  $w$  in practice since these can be computed on the fly using hash functions. The inner product approximation also yields some theoretical guarantees for the hash sampler.<sup>8</sup>

<sup>6</sup>We set  $l = 32$  bits in our experiments.

<sup>7</sup>In practice, we can ignore the *norm* terms to further speed up sampling since this is only an estimate for the proposal distribution and we follow this with the Metropolis Hastings step.

<sup>8</sup>For further details, please refer to (Ahmed et al., 2012).

## 4.1 Metropolis Hastings

In each sampling step, we use the distribution from Equation 10 as a proposal distribution in a Metropolis Hastings scheme to sample target translations for each source word.

Once a new target translation  $e'$  is sampled for source word  $f_i$  from the proposal distribution  $q(\cdot) \propto \exp^{s^l(f_i, e')}$ , we accept the proposal (and update the corresponding hash signatures) according to the probability  $r$

$$r = \frac{q(e_i^{old}) \cdot p_{new}(\cdot)}{q(e_i^{new}) \cdot p_{old}(\cdot)} \quad (11)$$

where,  $p_{old}(\cdot), p_{new}(\cdot)$  are the true conditional likelihood probabilities according to our model (including the language model component) for the old, new sample respectively.

## 5 Training Algorithm

Putting together all the pieces described in the previous section, we perform the following steps:

1. *Initialization:* We initialize the starting sample as follows: for each source word token, randomly sample a target word. If the source word also exists in the target vocabulary, then choose identity translation instead of the random one.<sup>9</sup>

2. *Hash Sampling Steps:* For each source word token  $f_i$ , run the hash sampler:

(a) Generate a proposal distribution by computing the hamming distance between the feature vectors for the source word and each target translation candidate. Sample a new target translation  $e_i$  for  $f_i$  from this distribution.

(b) Compute the acceptance probability for the chosen translation using a Metropolis Hastings scheme and accept (or reject) the sample. In practice, computation of the acceptance probability only needs to be done every  $r$  iterations (where  $r$  can be anywhere from 5 or 100).

Iterate through steps (2a) and (2b) for every word in the source text and then repeat this process for multiple iterations (usually 1000).

3. *Other Sampling Operators:* After every  $k$  iterations,<sup>10</sup> perform the following sampling operations:

(a) Re-ordering: For each source word token  $f_i$  at position  $i$ , randomly choose another position  $j$

<sup>9</sup>Initializing with identity translation rather than random choice helps in some cases, especially for *unknown* words that involve named entities, etc.

<sup>10</sup>We set  $k = 3$  in our experiments.

Corpus	Language	Sent.	Words	Vocab.
OPUS	Spanish	13,181	39,185	562
	English	19,770	61,835	411
EMEA	French	550,000	8,566,321	41,733
	Spanish	550,000	7,245,672	67,446

Table 1: Statistics of non-parallel corpora used here.

in the source sentence and swap the translations  $e_i$  with  $e_j$ . During the sampling process, we compute the probabilities for the two samples—the original and the swapped versions, and then sample an alignment from this distribution.

(b) Deletion: For each source word token, delete the current target translation (i.e., align it with the target NULL token). As with the re-ordering operation, we sample from a distribution consisting of the original and the deleted versions.

4. *Decoding the foreign sentence*: Finally, once the training is done (i.e., after all sampling iterations) we choose the final sample as our target translation output for the source text.

## 6 Experiments and Results

We test our method on two different corpora. To evaluate translation quality, we use BLEU score (Papineni et al., 2002), a standard evaluation measure used in machine translation.

First, we present MT results on non-parallel Spanish/English data from the OPUS corpus (Tiedemann, 2009) which was used by Ravi and Knight (2011b) and Nuhn et al. (2012). We show that our method achieves the best performance (BLEU scores) on this task while being significantly faster than both the previous approaches. We then apply our method to a much larger non-parallel French/Spanish corpus constructed from the EMEA corpus (Tiedemann, 2009). Here the vocabulary sizes are much larger and we show how our new Bayesian decipherment method scales well to this task in spite of using complex translation models. We also report the first BLEU results on such a large-scale MT task under truly non-parallel settings (without using any parallel data or seed lexicon).

For both the MT tasks, we also report BLEU scores for a baseline system using *identity* translations for common words (words appearing in both source/target vocabularies) and random translations for other words.

### 6.1 MT Task and Data

**OPUS movie subtitle corpus** (Tiedemann, 2009): This is a large open source collection of parallel corpora available for multiple language pairs. We use the same non-parallel Spanish/English corpus used in previous works (Ravi and Knight, 2011b; Nuhn et al., 2012). The details of the corpus are listed in Table 1. We use the entire Spanish source text for decipherment training and evaluate the final English output to report BLEU scores.

**EMEA corpus** (Tiedemann, 2009): This is a parallel corpus made out of PDF documents (articles from the medical domain) from the European Medicines Agency. We reserve the first 1k sentences in French as our source text (also used in decipherment training). To construct a non-parallel corpus, we split the remaining 1.1M lines as follows: first 550k sentences in French, last 550k sentences in Spanish. The latter is used to construct a target language model used for decipherment training. The corpus statistics are shown in Table 1.

### 6.2 Results

**OPUS**: We compare the MT results (BLEU scores) from different systems on the OPUS corpus in Table 2. The first row displays baseline performance. The next three rows 1a–1c display performance achieved by two methods from Ravi and Knight (2011b). Rows 2a, 2b show results from the of Nuhn et al. (2012). The last two rows display results for the new method using Bayesian hash sampling. Overall, using a 3-gram language model (instead of 2-gram) for decipherment training improves the performance for all methods. We observe that our method produces much better results than the others even with a 2-gram LM. With a 3-gram LM, the new method achieves the best performance; the highest BLEU score reported on this task. It is also interesting to note that the hash sampling method yields much better results than the Bayesian inference method presented in (Ravi and Knight, 2011b). This is due to the accelerated sampling scheme introduced earlier which helps it converge to better solutions faster.

Table 2 (last column) also compares the efficiency of different methods in terms of CPU time required for training. Both our 2-gram and 3-gram based methods are significantly faster than those previously reported for EM based training methods presented in (Ravi and Knight, 2011b; Nuhn

Method	BLEU	Time (hours)
Baseline system ( <i>identity translations</i> )	6.9	
1a. EM with 2-gram LM (Ravi and Knight, 2011b)	15.3	~850h
1b. EM with whole-segment LM (Ravi and Knight, 2011b)	19.3	
1c. Bayesian IBM Model 3 with 2-gram LM (Ravi and Knight, 2011b)	15.1	
2a. EM+Context with 2-gram LM (Nuhn et al., 2012)	15.2	50h
2b. EM+Context with 3-gram LM (Nuhn et al., 2012)	20.9	200h
3. Bayesian (standard) Gibbs sampling with 2-gram LM	-	222h
4a. Bayesian Hash Sampling* with 2-gram LM ( <i>this work</i> )	<b>20.3</b>	2.6h
4b. Bayesian Hash Sampling* with 3-gram LM ( <i>this work</i> ) (*sampler was run for 1000 iterations)	<b>21.2</b>	2.7h

Table 2: Comparison of MT performance (BLEU scores) and efficiency (running time in CPU hours) on the Spanish/English OPUS corpus using only non-parallel corpora for training. For the Bayesian methods 4a and 4b, the samplers were run for 1000 iterations each on a single machine (1.8GHz Intel processor). For 1a, 2a, 2b, we list the training times as reported by Nuhn et al. (2012) based on their EM implementation for different settings.

Method	BLEU
Baseline system ( <i>identity translations</i> )	3.0
Bayesian Hash Sampling with 2-gram LM vocab= <i>full</i> ( $V_e$ ), add_fertility= <i>no</i>	4.2
vocab= <i>pruned</i> *, add_fertility= <i>yes</i>	<b>5.3</b>

Table 3: MT results on the French/Spanish EMEA corpus using the new hash sampling method. \*The last row displays results when we sample target translations from a *pruned* candidate set (most frequent 1k Spanish words + identity translation candidates) which enables the sampler to run much faster when using more complex models.

et al., 2012). This is very encouraging since Nuhn et al. (2012) reported obtaining a speedup by pruning translation candidates (to  $\sim 1/8$ th the original size) prior to EM training. On the other hand, we sample from the full set of translation candidates including additional target phrase (of size 2) candidates which results in a much larger vocabulary consisting of 1600 candidates ( $\sim 4$  times the original size), yet our method runs much faster and yields better results. The table also demonstrates the significant speedup achieved by the hash sampler over a standard Gibbs sampler for the same model ( $\sim 85$  times faster when using a 2-gram LM).

We also compare the results against MT performance from *parallel training*—MOSES system (Koehn et al., 2007) trained on 20k sentence pairs. The comparable number for Table 2 is 63.6 BLEU.

Spanish ( $e$ )		French ( $f$ )
el	→	les
la	→	la
por	→	des
sección	→	rubrique
administración	→	administration

Table 4: Sample (1-best) Spanish/French translations produced by the new method on the EMEA corpus using word translation models trained with non-parallel corpora.

**EMEA Results** Table 3 shows the results achieved by our method on the larger task involving EMEA corpus. Here, the target vocabulary  $V_e$  is much higher (67k). In spite of this challenge and the model complexity, we can still perform decipherment training using Bayesian inference. We report the first BLEU score results on such a large-scale task using a 2-gram LM. This is achieved without using any seed lexicon or parallel corpora. The results are encouraging and demonstrates the ability of the method to scale to large-scale settings while performing efficient inference with complex models, which we believe will be especially useful for future MT application in scenarios where parallel data is hard to obtain. Table 4 displays some sample 1-best translations learned using this method.

For comparison purposes, we also evaluate MT performance on this task using *parallel training* (MOSES trained with hundred sentence pairs) and observe a BLEU score of 11.7.

## 7 Discussion and Future Work

There exists some work (Dou and Knight, 2012; Klementiev et al., 2012) that uses monolingual corpora to induce phrase tables, etc. These when combined with standard MT systems such as Moses (Koehn et al., 2007) trained on parallel corpora, have been shown to yield some BLEU score improvements. Nuhn et al. (2012) show some sample English/French lexicon entries learnt using EM algorithm with a pruned translation candidate set on a portion of the Gigaword corpus<sup>11</sup> but do not report any actual MT results. In addition, as we showed earlier our method can use Bayesian inference (which has a lot of nice properties compared to EM for unsupervised natural language tasks (Johnson, 2007; Goldwater and Griffiths, 2007)) and still scale easily to large vocabulary, data sizes while allowing the models to grow in complexity. Most importantly, our method produces better translation results (as demonstrated on the OPUS MT task). And to our knowledge, this is the first time that anyone has reported MT results under truly non-parallel settings on such a large-scale task (EMEA).

Our method is also easily extensible to out-of-domain translation scenarios similar to (Dou and Knight, 2012). While their work also uses Bayesian inference with a slice sampling scheme, our new approach uses a novel hash sampling scheme for decipherment that can easily scale to more complex models. The new decipherment framework also allows one to easily incorporate additional information (besides standard word translations) as features (e.g., context features, topic features, etc.) for unsupervised machine translation which can help further improve the performance in addition to accelerating the sampling process. We already demonstrated the utility of this system by going beyond words and incorporating phrase translations in a decipherment model for the first time.

In the future, we can obtain further speedups (especially for large-scale tasks) by parallelizing the sampling scheme seamlessly across multiple machines and CPU cores. The new framework can also be stacked with complementary techniques such as slice sampling, blocked (and type) sampling to further improve inference efficiency.

<sup>11</sup><http://www ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2003T05>

## 8 Conclusion

To summarize, our method is significantly faster than previous methods based on EM or Bayesian with standard Gibbs sampling and obtains better results than any previously published methods for the same task. The new framework also allows performing Bayesian inference for decipherment applications with more complex models than previously shown. We believe this framework will be useful for further extending MT models in the future to improve translation performance and for many other unsupervised decipherment application scenarios.

## References

- Amr Ahmed, Sujith Ravi, Shравan Narayanamurthy, and Alex Smola. 2012. Fastex: Hash clustering with exponential families. In *Proceedings of the 26th Conference on Neural Information Processing Systems (NIPS)*.
- Moses S. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thirty-fourth annual ACM Symposium on Theory of Computing*, pages 380–388.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.
- Qing Dou and Kevin Knight. 2012. Large scale decipherment for out-of-domain machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 266–275.
- Pascale Fung and Kathleen McKeown. 1997. Finding terminology translations from non-parallel corpora. In *Proceedings of the 5th Annual Workshop on Very Large Corpora*, pages 192–202.
- Sharon Goldwater and Tom Griffiths. 2007. A fully bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 744–751.
- Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL: HLT*, pages 771–779.
- Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 296–305.

- Alex Klementiev, Ann Irvine, Chris Callison-Burch, and David Yarowsky. 2012. Toward statistical machine translation without parallel corpora. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.
- Kevin Knight and Kenji Yamada. 1999. A computational approach to deciphering unknown scripts. In *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, pages 37–44.
- Philipp Koehn and Kevin Knight. 2000. Estimating word translation probabilities from unrelated monolingual corpora using the em algorithm. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 711–715.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180.
- Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 156–164.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318.
- Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 320–322.
- Sujith Ravi and Kevin Knight. 2011a. Bayesian inference for zodiac and other homophonic ciphers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 239–247.
- Sujith Ravi and Kevin Knight. 2011b. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21.
- Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 622–629.
- Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1048–1057.
- Jörg Tiedemann. 2009. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248.

# Automatic Interpretation of the English Possessive

Stephen Tratz<sup>†</sup>

Army Research Laboratory  
Adelphi Laboratory Center  
2800 Powder Mill Road  
Adelphi, MD 20783

stephen.c.tratz.civ@mail.mil

Eduard Hovy<sup>†</sup>

Carnegie Mellon University  
Language Technologies Institute  
5000 Forbes Avenue  
Pittsburgh, PA 15213

hovy@cmu.edu

## Abstract

The English 's possessive construction occurs frequently in text and can encode several different semantic relations; however, it has received limited attention from the computational linguistics community. This paper describes the creation of a semantic relation inventory covering the use of 's, an inter-annotator agreement study to calculate how well humans can agree on the relations, a large collection of possessives annotated according to the relations, and an accurate automatic annotation system for labeling new examples. Our 21,938 example dataset is by far the largest annotated possessives dataset we are aware of, and both our automatic classification system, which achieves 87.4% accuracy in our classification experiment, and our annotation data are publicly available.

## 1 Introduction

The English 's possessive construction occurs frequently in text—approximately 1.8 times for every 100 hundred words in the Penn Treebank<sup>1</sup> (Marcus et al., 1993)—and can encode a number of different semantic relations including ownership (*John's car*), part-of-whole (*John's arm*), extent (*6 hours' drive*), and location (*America's rivers*). Accurate automatic possessive interpretation could aid many natural language processing (NLP) applications, especially those that build semantic representations for text understanding, text generation, question answering, or information extraction. These interpretations could be valuable for machine translation to or from languages that allow different semantic relations to be encoded by

<sup>†</sup>The authors were affiliated with the USC Information Sciences Institute at the time this work was performed.

the possessive/genitive.

This paper presents an inventory of 17 semantic relations expressed by the English 's-construction, a large dataset annotated according to the this inventory, and an accurate automatic classification system. The final inter-annotator agreement study achieved a strong level of agreement, 0.78 Fleiss' Kappa (Fleiss, 1971) and the dataset is easily the largest manually annotated dataset of possessive constructions created to date. We show that our automatic classification system is highly accurate, achieving 87.4% accuracy on a held-out test set.

## 2 Background

Although the linguistics field has devoted significant effort to the English possessive (§6.1), the computational linguistics community has given it limited attention. By far the most notable exception to this is the line of work by Moldovan and Badulescu (Moldovan and Badulescu, 2005; Badulescu and Moldovan, 2009), who define a taxonomy of relations, annotate data, calculate inter-annotator agreement, and perform automatic classification experiments. Badulescu and Moldovan (2009) investigate both 's-constructions and *of* constructions in the same context using a list of 36 semantic relations (including OTHER). They take their examples from a collection of 20,000 randomly selected sentences from Los Angeles Times news articles used in TREC-9. For the 960 extracted 's-possessive examples, only 20 of their semantic relations are observed, including OTHER, with 8 of the observed relations occurring fewer than 10 times. They report a 0.82 Kappa agreement (Siegel and Castellan, 1988) for the two computational semantics graduates who annotate the data, stating that this strong result “can be explained by the instructions the annotators received

<sup>1</sup>Possessive pronouns such as *his* and *their* are treated as 's constructions in this work.

prior to annotation and by their expertise in Lexical Semantics.”

Moldovan and Badulescu experiment with several different classification techniques. They find that their *semantic scattering* technique significantly outperforms their comparison systems with its F-measure score of 78.75. Their SVM system performs the worst with only 23.25% accuracy—surprisingly low, especially considering that 220 of the 960 ‘s examples have the same label.

Unfortunately, Badulescu and Moldovan (2009) have not publicly released their data<sup>2</sup>. Also, it is sometimes difficult to understand the meaning of the semantic relations, partly because most relations are only described by a single example and, to a lesser extent, because the bulk of the given examples are *of*-constructions. For example, why *President of Bolivia* warrants a SOURCE/FROM relation but *University of Texas* is assigned to LOCATION/SPACE is unclear. Their relations and provided examples are presented below in Table 1.

Relation	Examples
POSSESSION	Mary’s book
KINSHIP	Mary’s brother
PROPERTY	John’s coldness
AGENT	investigation of the crew
TEMPORAL	last year’s exhibition
DEPICTION-DEPICTED	a picture of my niece
PART-WHOLE	the girl’s mouth
CAUSE	death of cancer
MAKE/PRODUCE	maker of computer
LOCATION/SPACE	Univerity of Texas
SOURCE/FROM	President of Bolivia
TOPIC	museum of art
ACCOMPANIMENT	solution of the problem
EXPERIENCER	victim of lung disease
RECIPIENT	Josephine’s reward
ASSOCIATED WITH	contractors of shipyard
MEASURE	hundred (sp?) of dollars
THEME	acquisition of the holding
RESULT	result of the review
OTHER	state of emergency

Table 1: The 20 (out of an original 36) semantic relations observed by Badulescu and Moldovan (2009) along with their examples.

### 3 Dataset Creation

We created the dataset used in this work from three different sources, each representing a distinct genre—newswire, non-fiction, and fiction. Of the

<sup>2</sup>Email requests asking for relation definitions and the data were not answered, and, thus, we are unable to provide an informative comparison with their work.

21,938 total examples, 15,330 come from sections 2–21 of the Penn Treebank (Marcus et al., 1993). Another 5,266 examples are from *The History of the Decline and Fall of the Roman Empire* (Gibbon, 1776), a non-fiction work, and 1,342 are from *The Jungle Book* (Kipling, 1894), a collection of fictional short stories. For the Penn Treebank, we extracted the examples using the provided gold standard parse trees, whereas, for the latter cases, we used the output of an open source parser (Tratz and Hovy, 2011).

## 4 Semantic Relation Inventory

The initial semantic relation inventory for possessives was created by first examining some of the relevant literature on possessives, including work by Badulescu and Moldovan (2009), Barker (1995), Quirk et al. (1985), Rosenbach (2002), and Taylor (1996), and then manually annotating the large dataset of examples. Similar examples were grouped together to form initial categories, and groups that were considered more difficult were later reexamined in greater detail. Once all the examples were assigned to initial categories, the process of refining the definitions and annotations began.

In total, 17 relations were created, not including OTHER. They are shown in Table 3 along with approximate (best guess) mappings to relations defined by others, specifically those of Quirk et al. (1985), whose relations are presented in Table 2, as well as Badulescu and Moldovan’s (2009) relations.

Relation	Examples
POSSESSIVE	my wife’s father
SUBJECTIVE	boy’s application
OBJECTIVE	the family’s support
ORIGIN	the general’s letter
DESCRIPTIVE	a women’s college
MEASURE	ten days’ absense
ATTRIBUTE	the victim’s courage
PARTITIVE	the baby’s eye
APPOSITION (marginal)	Dublin’s fair city

Table 2: The semantic relations proposed by Quirk et al. (1985) for ‘s along with some of their examples.

### 4.1 Refinement and Inter-annotator Agreement

The semantic relation inventory was refined using an iterative process, with each iteration involv-

Relation	Example	HDFRE	JB	PTB	Mappings
SUBJECTIVE	Dora’s travels	1083	89	3169	Q:SUBJECTIVE, B:AGENT
PRODUCER’S PRODUCT	Ford’s Taurus	47	44	1183	Q:ORIGIN, B:MAKE/PRODUCE B:RESULT
OBJECTIVE	Mowgli’s capture	380	7	624	Q:OBJECTIVE, B:THEME
CONTROLLER/OWNER/USER	my apartment	882	157	3940	QB:POSSESSIVE
MENTAL EXPERIENCER	Sam’s fury	277	22	232	Q:POSSESSIVE, B:EXPERIENCER
RECIPIENT	their bonuses	12	6	382	Q:POSSESSIVE, B:RECIPIENT
MEMBER’S COLLECTION	John’s family	144	31	230	QB:POSSESSIVE
PARTITIVE	John’s arm	253	582	451	Q:PARTITIVE, B:PART-WHOLE
LOCATION	Libya’s people	24	0	955	Q:POSSESSIVE, B:SOURCE/FROM B:LOCATION/SPACE
TEMPORAL	today’s rates	0	1	623	Q:POSSESSIVE, B:TEMPORAL
EXTENT	6 hours’ drive	8	10	5	QB:MEASURE
KINSHIP	Mary’s kid	324	156	264	Q:POSSESSIVE, B:KINSHIP
ATTRIBUTE	picture’s vividness	1013	34	1017	Q:ATTRIBUTE, B:PROPERTY
TIME IN STATE	his years in Ohio	145	32	237	QB:POSSESSIVE
POSSESSIVE COMPOUND	the [men’s room]	0	0	67	Q:DESCRIPTIVE
ADJECTIVE DETERMINED	his fellow Brit	12	0	33	
OTHER RELATIONAL NOUN	his friend	629	112	1772	QB:POSSESSIVE
OTHER	your Lordship	33	59	146	B:OTHER

Table 3: Possessive semantic relations along with examples, counts, and approximate mappings to other inventories. Q and B represent Quirk et al. (1985) and Badulescu and Moldovan (2009), respectively. HDFRE, JB, PTB: *The History of the Decline and Fall of the Roman Empire*, *The Jungle Book*, and the Penn Treebank, respectively.

ing the annotation of a random set of 50 examples. Each set of examples was extracted such that no two examples had an identical *possessee* word. For a given example, annotators were instructed to select the most appropriate option but could also record a second-best choice to provide additional feedback. Figure 1 presents a screenshot of the HTML-based annotation interface. After the annotation was complete for a given round, agreement and entropy figures were calculated and changes were made to the relation definitions and dataset. The number of refinement rounds was arbitrarily limited to five. To measure agreement, in addition to calculating simple percentage agreement, we computed Fleiss’ Kappa (Fleiss, 1971), a measure of agreement that incorporates a correction for agreement due to chance, similar to Cohen’s Kappa (Cohen, 1960), but which can be used to measure agreement involving more than two annotations per item. The agreement and entropy figures for these five intermediate annotation rounds are given in Table 4. In all the possessive annotation tables, Annotator A refers to the primary author and the labels B and C refer to two additional annotators.

To calculate a final measure of inter-annotator agreement, we randomly drew 150 examples from the dataset not used in the previous refinement iterations, with 50 examples coming from each of

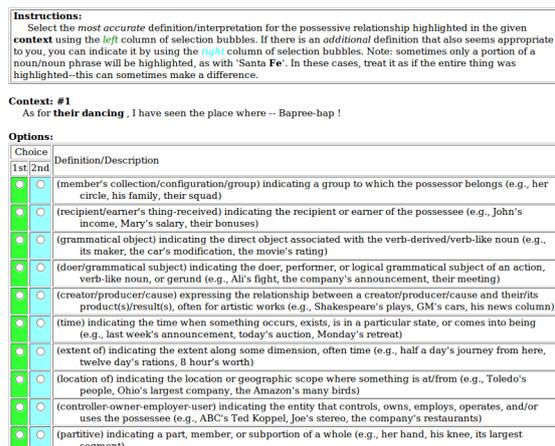


Figure 1: Screenshot of the HTML template page used for annotation.

the three original data sources. All three annotators initially agreed on 82 of the 150 examples, leaving 68 examples with at least some disagreement, including 17 for which all three annotators disagreed.

Annotators then engaged in a new task in which they re-annotated these 68 examples, in each case being able to select only from the definitions previously chosen for each example by at least one annotator. No indication of who or how many people had previously selected the definitions was

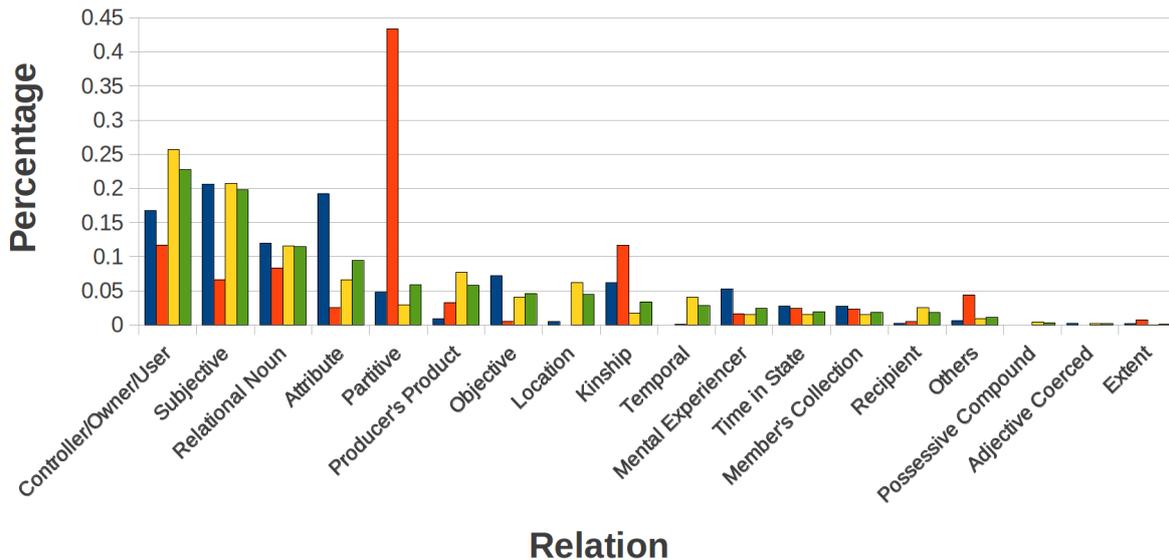


Figure 2: Semantic relation distribution for the dataset presented in this work. HDFRE: History of the Decline and Fall of the Roman Empire; JB: Jungle Book; PTB: Sections 2–21 of the Wall Street Journal portion of the Penn Treebank.

given<sup>3</sup>. Annotators were instructed not to choose a definition simply because they thought they had chosen it before or because they thought someone else had chosen it. After the revision process, all three annotators agreed in 109 cases and all three disagreed in only 6 cases. During the revision process, Annotator A made 8 changes, B made 20 changes, and C made 33 changes. Annotator A likely made the fewest changes because he, as the primary author, spent a significant amount of time thinking about, writing, and re-writing the definitions used for the various iterations. Annotator C’s annotation work tended to be less consistent in general than Annotator B’s throughout this work as well as in a different task not discussed within this paper, which probably why Annotator C made more changes than Annotator B. Prior to this revision process, the three-way Fleiss’ Kappa score was 0.60 but, afterwards, it was at 0.78. The inter-annotator agreement and entropy figures for before and after this revision process, including pairwise scores between individual annotators, are presented in Tables 5 and 6.

## 4.2 Distribution of Relations

The distribution of semantic relations varies somewhat by the data source. *The Jungle Book*’s distribution is significantly different from the oth-

<sup>3</sup>Of course, if three definitions were present, it could be inferred that all three annotators had initially disagreed.

ers, with a much larger percentage of PARTITIVE and KINSHIP relations. The Penn Treebank and *The History of the Decline and Fall of the Roman Empire* were substantially more similar, although there are notable differences. For instance, the LOCATION and TEMPORAL relations almost never occur in *The History of the Decline and Fall of the Roman Empire*. Whether these differences are due to variations in genre, time period, and/or other factors would be an interesting topic for future study. The distribution of relations for each data source is presented in Figure 2.

Though it is harder to compare across datasets using different annotation schemes, there are at least a couple notable differences between the distribution of relations for Badulescu and Moldovan’s (2009) dataset and the distribution of relations used in this work. One such difference is the much higher percentage of examples labeled as TEMPORAL—11.35% vs only 2.84% in our data. Another difference is a higher incidence of the KINSHIP relation (6.31% vs 3.39%), although it is far less frequent than it is in *The Jungle Book* (11.62%).

## 4.3 Encountered Ambiguities

One of the problems with creating a list of relations expressed by ’s-constructions is that some examples can potentially fit into multiple categories. For example, *Joe’s resentment* encodes

both SUBJECTIVE relation and MENTAL EXPERIENCER relations and *UK's cities* encodes both PARTITIVE and LOCATION relations. A representative list of these types of issues along with examples designed to illustrate them is presented in Table 7.

## 5 Experiments

For the automatic classification experiments, we set aside 10% of the data for test purposes, and used the the remaining 90% for training. We used 5-fold cross-validation performed using the training data to tweak the included feature templates and optimize training parameters.

### 5.1 Learning Approach

The LIBLINEAR (Fan et al., 2008) package was used to train linear Support Vector Machine (SVMs) for all the experiments in the one-against-the-rest style. All training parameters took their default values with the exception of the C parameter, which controls the tradeoff between margin width and training error and which was set to 0.02, the point of highest performance in the cross-validation tuning.

### 5.2 Feature Generation

For feature generation, we conflated the possessive pronouns ‘his’, ‘her’, ‘my’, and ‘your’ to ‘person.’ Similarly, every term matching the case-insensitive regular expression `(corp|col|plc|in|clag|ltd|llc)\.?` was replaced with the word ‘corporation.’

All the features used are functions of the following five words.

- The possessor word
- The possessee word
- The syntactic governor of the possessee word
- The set of words between the possessor and possessee word (e.g., *first* in *John's first kiss*)
- The word to the right of the possessee

The following feature templates are used to generate features from the above words. Many of these templates utilize information from WordNet (Fellbaum, 1998).

- WordNet link types (link type list) (e.g., attribute, hypernym, entailment)
- Lexicographer filenames (lexnames)—top level categories used in WordNet (e.g., noun.body, verb.cognition)

- Set of words from the WordNet definitions (gloss terms)
- The list of words connected via WordNet *part-of* links (part words)
- The word's text (the word itself)
- A collection of affix features (e.g., -ion, -er, -ity, -ness, -ism)
- The last {2,3} letters of the word
- List of all possible parts-of-speech in WordNet for the word
- The part-of-speech assigned by the part-of-speech tagger
- WordNet hypernyms
- WordNet synonyms
- Dependent words (all words linked as children in the parse tree)
- Dependency relation to the word's syntactic governor

### 5.3 Results

The system predicted correct labels for 1,962 of the 2,247 test examples, or 87.4%. The accuracy figures for the test instances from the Penn Treebank, *The Jungle Book*, and *The History of the Decline and Fall of the Roman Empire* were 88.8%, 84.7%, and 80.6%, respectively. The fact that the score for *The Jungle Book* was the lowest is somewhat surprising considering it contains a high percentage of body part and kinship terms, which tend to be straightforward, but this may be because the other sources comprise approximately 94% of the training examples.

Given that human agreement typically represents an upper bound on machine performance in classification tasks, the 87.4% accuracy figure may be somewhat surprising. One explanation is that the examples pulled out for the inter-annotator agreement study each had a unique *possessee* word. For example, “expectations”, as in “analyst's expectations”, occurs 26 times as the possessee in the dataset, but, for the inter-annotator agreement study, at most one of these examples could be included. More importantly, when the initial relations were being defined, the data were first sorted based upon the possessee and then the possessor in order to create blocks of similar examples. Doing this allowed multiple examples to be assigned to a category more quickly because one can decide upon a category for the whole lot at once and then just extract the few, if any, that belong to other categories. This is likely to be both faster and more consistent than examining each

Iteration	Agreement (%)			Fleiss' $\kappa$				Entropy		
	A vs B	A vs C	B vs C	A vs B	A vs C	B vs C	All	A	B	C
1	0.60	0.68	0.54	0.53	0.62	0.46	0.54	3.02	2.98	3.24
2	0.64	0.44	0.50	0.59	0.37	0.45	0.47	3.13	3.40	3.63
3	0.66	0.66	0.72	0.57	0.58	0.66	0.60	2.44	2.47	2.70
4	0.64	0.30	0.38	0.57	0.16	0.28	0.34	2.80	3.29	2.87
5	0.72	0.66	0.60	0.67	0.61	0.54	0.61	3.21	3.12	3.36

Table 4: Intermediate results for the possessives refinement work.

Portion	Agreement (%)			Fleiss' $\kappa$				Entropy		
	A vs B	A vs C	B vs C	A vs B	A vs C	B vs C	All	A	B	C
PTB	0.62	0.62	0.54	0.56	0.56	0.46	0.53	3.22	3.17	3.13
HDFRE	0.82	0.78	0.72	0.77	0.71	0.64	0.71	2.73	2.75	2.73
JB	0.74	0.56	0.54	0.70	0.50	0.48	0.56	3.17	3.11	3.17
All	0.73	0.65	0.60	0.69	0.61	0.55	0.62	3.43	3.35	3.51

Table 5: Final possessives annotation agreement figures before revisions.

Source	Agreement (%)			Fleiss' $\kappa$				Entropy		
	A vs B	A vs C	B vs C	A vs B	A vs C	B vs C	All	A	B	C
PTB	0.78	0.74	0.74	0.75	0.70	0.70	0.72	3.30	3.11	3.35
HDFRE	0.78	0.76	0.76	0.74	0.72	0.72	0.73	3.03	2.98	3.17
JB	0.92	0.90	0.86	0.90	0.87	0.82	0.86	2.73	2.71	2.65
All	0.83	0.80	0.79	0.80	0.77	0.76	0.78	3.37	3.30	3.48

Table 6: Final possessives annotation agreement figures after revisions.

First Relation	Second Relation	Example
PARTITIVE	CONTROLLER/...	<i>BoA's Mr. Davis</i>
PARTITIVE	LOCATION	<i>UK's cities</i>
PARTITIVE	OBJECTIVE	<i>BoA's adviser</i>
PARTITIVE	OTHER RELATIONAL NOUN	<i>BoA's chairman</i>
PARTITIVE	PRODUCER'S PRODUCT	<i>the lamb's wool</i>
CONTROLLER/...	PRODUCER'S PRODUCT	<i>the bird's nest</i>
CONTROLLER/...	OBJECTIVE	<i>his assistant</i>
CONTROLLER/...	LOCATION	<i>Libya's oil company</i>
CONTROLLER/...	ATTRIBUTE	<i>Joe's strength</i>
CONTROLLER/...	MEMBER'S COLLECTION	<i>the colonel's unit</i>
CONTROLLER/...	RECIPIENT	<i>Joe's trophy</i>
RECIPIENT	OBJECTIVE	<i>Joe's reward</i>
SUBJECTIVE	PRODUCER'S PRODUCT	<i>Joe's announcement</i>
SUBJECTIVE	OBJECTIVE	<i>its change</i>
SUBJECTIVE	CONTROLLER/...	<i>Joe's employee</i>
SUBJECTIVE	LOCATION	<i>Libya's devolution</i>
SUBJECTIVE	MENTAL EXPERIENCER	<i>Joe's resentment</i>
OBJECTIVE	MENTAL EXPERIENCER	<i>Joe's concern</i>
OBJECTIVE	LOCATION	<i>the town's inhabitants</i>
KINSHIP	OTHER RELATIONAL NOUN	<i>his fiancée</i>

Table 7: Ambiguous/multiclass possessive examples.

example in isolation. This advantage did not exist in the inter-annotator agreement study.

#### 5.4 Feature Ablation Experiments

To evaluate the importance of the different types of features, the same experiment was re-run multiple times, each time including or excluding exactly one feature template. Before each variation, the C

parameter was retuned using 5-fold cross validation on the training data. The results for these runs are shown in Table 8.

Based upon the leave-one-out and only-one feature evaluation experiment results, it appears that the possessee word is more important to classification than the possessor word. The possessor word is still valuable though, with it likely being more

valuable for certain categories (e.g., TEMPORAL and LOCATION) than others (e.g., KINSHIP). Hypernym and gloss term features proved to be about equally valuable. Curiously, although hypernyms are commonly used as features in NLP classification tasks, gloss terms, which are rarely used for these tasks, are approximately as useful, at least in this particular context. This would be an interesting result to examine in greater detail.

## 6 Related Work

### 6.1 Linguistics

Semantic relation inventories for the English 's-construction have been around for some time; Taylor (1996) mentions a set of 6 relations enumerated by Poutsma (1914–1916). Curiously, there is not a single dominant semantic relation inventory for possessives. A representative example of semantic relation inventories for 's-constructions is the one given by Quirk et al. (1985) (presented earlier in Section 2).

Interestingly, the set of relations expressed by possessives varies by language. For example, Classical Greek permits a *standard of comparison* relation (e.g., “better than Plato”) (Nikiforidou, 1991), and, in Japanese, some relations are expressed in the opposite direction (e.g., “blue eye’s doll”) while others are not (e.g., “Tanaka’s face”) (Nishiguchi, 2009).

To explain how and why such seemingly different relations as *whole+part* and *cause+effect* are expressed by the same linguistic phenomenon, Nikiforidou (1991) pursues an approach of *metaphorical structuring* in line with the work of Lakoff and Johnson (1980) and Lakoff (1987). She thus proposes a variety of such metaphors as THINGS THAT HAPPEN (TO US) ARE (OUR) POSSESSIONS and CAUSES ARE ORIGINS to explain how the different relations expressed by possessives extend from one another.

Certainly, not all, or even most, of the linguistics literature on English possessives focuses on creating lists of semantic relations. Much of the work covering the semantics of the 's construction in English, such as Barker’s (1995) work, dwells on the split between cases of *relational* nouns, such as *sister*, that, by their very definition, hold a specific relation to other real or conceptual things, and *non-relational*, or *sortal* nouns (Löbner, 1985), such as *car*.

Vikner and Jensen’s (2002) approach for han-

dling these disparate cases is based upon Pustejovsky’s (1995) generative lexicon framework. They coerce sortal nouns (e.g., *car*) into being relational, purporting to create a uniform analysis. They split *lexical possession* into four types: *inherent*, *part-whole*, *agentive*, and *control*, with *agentive* and *control* encompassing many, if not most, of the cases involving sortal nouns.

A variety of other issues related to possessives considered by the linguistics literature include adjectival modifiers that significantly alter interpretation (e.g., *favorite* and *former*), double genitives (e.g., *book of John’s*), bare possessives (i.e., cases where the possessee is omitted, as in “Eat at Joe’s”), possessive compounds (e.g., *driver’s license*), the syntactic structure of possessives, definitiveness, changes over the course of history, and differences between languages in terms of which relations may be expressed by the genitive. Representative work includes that by Barker (1995), Taylor (1996), Heine (1997), Partee and Borschev (1998), Rosenbach (2002), and Vikner and Jensen (2002).

### 6.2 Computational Linguistics

Though the relation between nominals in the English possessive construction has received little attention from the NLP community, there is a large body of work that focuses on similar problems involving noun-noun relation interpretation/paraphrasing, including interpreting the relations between the components of noun compounds (Butnariu et al., 2010), disambiguating preposition senses (Litkowski and Hargraves, 2007), or annotating the relation between nominals in more arbitrary constructions within the same sentence (Hendrickx et al., 2009).

Whereas some of these lines of work use fixed inventories of semantic relations (Lauer, 1995; Nastase and Szpakowicz, 2003; Kim and Baldwin, 2005; Girju, 2009; Ó Séaghdha and Copestake, 2009; Tratz and Hovy, 2010), other work allows for a nearly infinite number of interpretations (Butnariu and Veale, 2008; Nakov, 2008). Recent SemEval tasks (Butnariu et al., 2009; Hendrickx et al., 2013) pursue this more open-ended strategy. In these tasks, participating systems recover the implicit predicate between the nouns in noun compounds by creating potentially unique paraphrases for each example. For instance, a system might generate the paraphrase *made of* for the noun com-

Feature Type	Word(s)						Results	
	L	R	C	G	B	N	LOO	OO
Gloss Terms		■					0.867 (0.04)	0.762 (0.08)
Hypernyms		■					0.870 (0.04)	0.760 (0.16)
Synonyms		■					0.873 (0.04)	0.757 (0.32)
Word Itself		■					0.871 (0.04)	0.745 (0.08)
Lexnames		■					0.871 (0.04)	0.514 (0.32)
Last Letters		■					0.870 (0.04)	0.495 (0.64)
Lexnames			■				0.872 (0.04)	0.424 (0.08)
Link types		■					0.874 (0.02)	0.398 (0.64)
Link types			■				0.870 (0.04)	0.338 (0.32)
Word Itself	■						0.870 (0.04)	0.316 (0.16)
Last Letters	■						0.872 (0.02)	0.303 (0.16)
Gloss Terms	■						0.872 (0.02)	0.271 (0.04)
Hypernyms	■						0.875 (0.02)	0.269 (0.08)
Word Itself						■	0.874 (0.02)	0.261 (0.08)
Synonyms	■						0.874 (0.02)	0.260 (0.04)
Lexnames	■						0.874 (0.02)	0.247 (0.04)
Part-of-speech List			■				0.873 (0.02)	0.245 (0.16)
Part-of-speech List		■					0.874 (0.02)	0.243 (0.16)
Dependency		■					0.872 (0.02)	0.241 (0.16)
Part-of-speech List	■						0.874 (0.02)	0.236 (0.32)
Link Types	■						0.874 (0.02)	0.236 (0.64)
Word Itself					■		0.870 (0.02)	0.234 (0.32)
Assigned Part-of-Speech		■					0.874 (0.02)	0.228 (0.08)
Affixes		■					0.873 (0.02)	0.227 (0.16)
Assigned Part-of-Speech					■		0.873 (0.02)	0.194 (0.16)
Hypernyms				■			0.873 (0.02)	0.186 (0.04)
Lexnames				■			0.870 (0.04)	0.170 (0.64)
Text of Dependents		■					0.874 (0.02)	0.156 (0.08)
Parts List		■					0.873 (0.02)	0.141 (0.16)
Affixes	■						0.870 (0.04)	0.114 (0.32)
Affixes			■				0.873 (0.02)	0.105 (0.04)
Parts List	■						0.874 (0.02)	0.103 (0.16)

Table 8: Results for leave-one-out and only-one feature template ablation experiment results for all feature templates sorted by the only-one case. L, R, C, G, B, and N stand for *left word (possessor)*, *right word (possessee)*, *pairwise combination of outputs for possessor and possessee*, *syntactic governor of possessee*, *all tokens between possessor and possessee*, and *the word next to the possessee (on the right)*, respectively. The C parameter value used to train the SVMs is shown in parentheses.

pound *pepperoni pizza*. Computer-generated results are scored against a list of human-generated options in order to rank the participating systems. This approach could be applied to possessives interpretation as well.

Concurrent with the lack of NLP research on the subject is the absence of available annotated datasets for training, evaluation, and analysis. The NomBank project (Meyers et al., 2004) provides coarse annotations for some of the possessive constructions in the Penn Treebank, but only those that meet their criteria.

## 7 Conclusion

In this paper, we present a semantic relation inventory for 's possessives consisting of 17 relations expressed by the English 's construction, the

largest available manually-annotated collection of possessives, and an effective method for automatically assigning the relations to unseen examples. We explain our methodology for building this inventory and dataset and report a strong level of inter-annotator agreement, reaching 0.78 Kappa overall. The resulting dataset is quite large, at 21,938 instances, and crosses multiple domains, including news, fiction, and historical non-fiction. It is the only large fully-annotated publicly-available collection of possessive examples that we are aware of. The straightforward SVM-based automatic classification system achieves 87.4% accuracy—the highest automatic possessive interpretation accuracy figured reported to date. These high results suggest that SVMs are a good choice for automatic possessive interpre-

tation systems, in contrast to Moldovan and Badulescu (2005) findings. The data and software presented in this paper are available for download at <http://www.isi.edu/publications/licensed-sw/fanparser/index.html>.

## 8 Future Work

Going forward, we would like to examine the various ambiguities of possessives described in Section 4.3. Instead of trying to find the one-best interpretation for a given possessive example, we would like to produce a list of *all* appropriate interpretations.

Another avenue for future research is to study variation in possessive use across genres, including scientific and technical genres. Similarly, one could automatically process large volumes of text from various time periods to investigate changes in the use of the possessive over time.

## Acknowledgments

We would like to thank Charles Zheng and Sarah Benzel for all their annotation work and valuable feedback.

## References

- Adriana Badulescu and Dan Moldovan. 2009. A Semantic Scattering Model for the Automatic Interpretation of English Genitives. *Natural Language Engineering*, 15:215–239.
- Chris Barker. 1995. *Possessive Descriptions*. CSLI Publications, Stanford, CA, USA.
- Cristina Butnariu and Tony Veale. 2008. A Concept-Centered Approach to Noun-Compound Interpretation. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 81–88.
- Cristina Butnariu, Su Nam Kim, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2009. SemEval-2010 Task 9: The Interpretation of Noun Compounds Using Paraphrasing Verbs and Prepositions. In *DEW '09: Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 100–105.
- Cristina Butnariu, Su Nam Kim, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2010. SemEval-2010 Task 9: The Interpretation of Noun Compounds Using Paraphrasing Verbs and Prepositions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 39–44.
- Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874.
- Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, MA.
- Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.
- Edward Gibbon. 1776. *The History of the Decline and Fall of the Roman Empire*, volume I of *The History of the Decline and Fall of the Roman Empire*. Printed for W. Strahan and T. Cadell.
- Roxanna Girju. 2009. The Syntax and Semantics of Prepositions in the Task of Automatic Interpretation of Nominal Phrases and Compounds: A Cross-linguistic Study. *Computational Linguistics - Special Issue on Prepositions in Application*, 35(2):185–228.
- Bernd Heine. 1997. *Possession: Cognitive Sources, Forces, and Grammaticalization*. Cambridge University Press, United Kingdom.
- Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009)*, pages 94–99.
- Iris Hendrickx, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2013. Task Description: SemEval-2013 Task 4: Free Paraphrases of Noun Compounds. <http://www.cs.york.ac.uk/semeval-2013/task4/>. [Online; accessed 1-May-2013].
- Su Nam Kim and Timothy Baldwin. 2005. Automatic Interpretation of Noun Compounds using WordNet::Similarity. *Natural Language Processing-IJCNLP 2005*, pages 945–956.
- Rudyard Kipling. 1894. *The Jungle Book*. Macmillan, London, UK.
- George Lakoff and Mark Johnson. 1980. *Metaphors We Live by*. The University of Chicago Press, Chicago, USA.
- George Lakoff. 1987. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. The University of Chicago Press, Chicago, USA.
- Mark Lauer. 1995. Corpus Statistics Meet the Noun Compound: Some Empirical Results. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 47–54.
- Ken Litkowski and Orin Hargraves. 2007. SemEval-2007 Task 06: Word-Sense Disambiguation of Prepositions. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 24–29.

- Sebastian Lbner. 1985. Definites. *Journal of Semantics*, 4(4):279.
- Mitchell P. Marcus, Mary A. Marcinkiewicz, and Beatrice Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):330.
- Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank Project: An Interim Report. In *Proceedings of the NAACL/HLT Workshop on Frontiers in Corpus Annotation*.
- Dan Moldovan and Adriana Badulescu. 2005. A Semantic Scattering Model for the Automatic Interpretation of Genitives. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 891–898.
- Preslav Nakov. 2008. Noun Compound Interpretation Using Paraphrasing Verbs: Feasibility Study. In *Proceedings of the 13th International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, pages 103–117.
- Vivi Nastase and Stan Szpakowicz. 2003. Exploring Noun-Modifier Semantic Relations. In *Fifth International Workshop on Computational Semantics (IWCS-5)*, pages 285–301.
- Kiki Nikiforidou. 1991. The Meanings of the Genitive: A Case Study in the Semantic Structure and Semantic Change. *Cognitive Linguistics*, 2(2):149–206.
- Sumiyo Nishiguchi. 2009. Qualia-Based Lexical Knowledge for the Disambiguation of the Japanese Postposition No. In *Proceedings of the Eighth International Conference on Computational Semantics*.
- Diarmuid Ó Saghdha and Ann Copestake. 2009. Using Lexical and Relational Similarity to Classify Semantic Relations. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 621–629.
- Barbara H. Partee and Vladimir Borschev. 1998. Integrating Lexical and Formal Semantics: Genitives, Relational Nouns, and Type-Shifting. In *Proceedings of the Second Tbilisi Symposium on Language, Logic, and Computation*, pages 229–241.
- James Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge, MA, USA.
- Randolph Quirk, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman Inc., New York.
- Anette Rosenbach. 2002. *Genitive Variation in English: Conceptual Factors in Synchronic and Diachronic Studies*. Topics in English linguistics. Mouton de Gruyter.
- Sidney Siegel and N. John Castellan. 1988. *Non-parametric statistics for the behavioral sciences*. McGraw-Hill.
- John R. Taylor. 1996. *Possessives in English*. Oxford University Press, New York.
- Stephen Tratz and Eduard Hovy. 2010. A Taxonomy, Dataset, and Classifier for Automatic Noun Compound Interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 678–687.
- Stephen Tratz and Eduard Hovy. 2011. A Fast, Accurate, Non-Projective, Semantically-Enriched Parser. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1268.
- Carl Vikner and Per Anker Jensen. 2002. A Semantic Analysis of the English Genitive. Interaction of Lexical and Formal Semantics. *Studia Linguistica*, 56(2):191–226.

# Is a 204 cm Man Tall or Small ?

## Acquisition of Numerical Common Sense from the Web

Katsuma Narisawa<sup>1</sup>    Yotaro Watanabe<sup>1</sup>    Junta Mizuno<sup>2</sup>  
Naoaki Okazaki<sup>1,3</sup>    Kentaro Inui<sup>1</sup>

<sup>1</sup>Graduate School of Information Sciences, Tohoku University

<sup>2</sup>National Institute of Information and Communications Technology (NICT)

<sup>3</sup>Japan Science and Technology Agency (JST)

{katsuma, yotaro-w, junta-m, okazaki, inui}@ecei.tohoku.ac.jp

### Abstract

This paper presents novel methods for modeling *numerical common sense*: the ability to infer whether a given number (e.g., *three billion*) is large, small, or normal for a given context (e.g., *number of people facing a water shortage*). We first discuss the necessity of numerical common sense in solving textual entailment problems. We explore two approaches for acquiring numerical common sense. Both approaches start with extracting numerical expressions and their *context* from the Web. One approach estimates the distribution of numbers co-occurring within a context and examines whether a given value is large, small, or normal, based on the distribution. Another approach utilizes textual patterns with which speakers explicitly express their judgment about the value of a numerical expression. Experimental results demonstrate the effectiveness of both approaches.

### 1 Introduction

Textual entailment recognition (RTE) involves a wide range of semantic inferences to determine whether the meaning of a hypothesis sentence ( $h$ ) can be inferred from another text ( $t$ ) (Dagan et al., 2006). Although several evaluation campaigns (e.g., PASCAL/TAC RTE challenges) have made significant progress, the RTE community recognizes the necessity of a deeper understanding of the core phenomena involved in textual inference. Such recognition comes from the ideas that crucial progress may derive from decomposing the complex RTE task into basic phenomena and from solving each basic phenomenon separately (Bentivogli et al., 2010; Sammons et al., 2010; Cabrio and Magnini, 2011; Toledo et al., 2012).

Given this background, we focus on solving one of the basic phenomena in RTE: semantic inference related to numerical expressions. The specific problem we address is acquisition of *numerical common sense*. For example,

(1)  $t$  : *Before long, 3b people will face a water shortage in the world.*

$h$  : *Before long, a serious water shortage will occur in the world.*

Although recognizing the entailment relation between  $t$  and  $h$  is frustratingly difficult, we assume this inference is decomposable into three phases:

3b people face a water shortage.

$\Leftrightarrow$  3,000,000,000 people face a water shortage.

$\models$  many people face a water shortage.

$\models$  a serious water shortage.

In the first phase, it is necessary to recognize  $3b$  as a numerical expression and to resolve the expression  $3b$  into the exact amount *3,000,000,000*. The second phase is much more difficult because we need subjective but common-sense knowledge that *3,000,000,000 people* is a large number.

In this paper, we address the first and second phases of inference as an initial step towards semantic processing with numerical expressions. The contributions of this paper are four-fold.

1. We examine instances in existing RTE corpora, categorize them into groups in terms of the necessary semantic inferences, and discuss the impact of this study for solving RTE problems with numerical expressions.
2. We describe a method of normalizing numerical expressions referring to the same amount in text into a unified semantic representation.
3. We present approaches for aggregating numerical common sense from examples of numerical expressions and for judging whether a given amount is large, small, or normal.

4. We demonstrate the effectiveness of this approach, reporting experimental results and analyses in detail. Although it would be ideal to evaluate the impact of this study on the overall RTE task, we evaluate each phase separately. We do this because the existing RTE data sets tend to exhibit very diverse linguistic phenomena, and it is difficult to employ such data for evaluating the real impact of this study.

## 2 Related work

Surprisingly, NLP research has paid little attention to semantic processing of numerical expressions. This is evident when we compare with temporal expressions, for which corpora (e.g., ACE-2005<sup>1</sup>, TimeBank<sup>2</sup>) were developed with annotation schemes (e.g., TIMEX<sup>3</sup>, TimeML<sup>4</sup>).

Several studies deal with numerical expressions in the context of information extraction (Bakalov et al., 2011), information retrieval (Fontoura et al., 2006; Yoshida et al., 2010), and question answering (Moriceau, 2006). Numbers such as product prices and weights have been common targets of information extraction. Fontoura et al. (2006) and Yoshida et al. (2010) presented algorithms and data structures that allow number-range queries for searching documents. However, these studies do not interpret the quantity (e.g., *3,000,000,000*) of a numerical expression (e.g., *3b people*), but rather treat numerical expressions as strings.

Banerjee et al. (2009) focused on quantity consensus queries, in which there is uncertainty about the quantity (e.g., *weight airbus A380 pounds*). Given a query, their approach retrieves documents relevant to the query and identifies the quantities of numerical expressions in the retrieved documents. They also proposed methods for enumerating and ranking the candidates for the consensus quantity intervals. Even though our study shares a similar spirit (modeling of consensus for quantities) with Banerjee et al. (2009), their goal is different: to determine ground-truth values for queries.

In question answering, to help “sanity check” answers with numerical values that were

way out of common-sense ranges, IBM’s PI-QUANT (Prager et al., 2003; Chu-Carroll et al., 2003) used information in Cyc (Lenat, 1995). For example, their question-answering system rejects *200 miles* as a candidate answer for the *height of Mt. Everest*, since Cyc knows mountains are between 1,000 and 30,000 ft. high. They also consider the problem of variations in the precision of numbers (e.g., *5 million*, *5.1 million*, *5,200,390*) and unit conversions (e.g., *square kilometers* and *acres*).

Some recent studies delve deeper into the semantic interpretation of numerical expressions. Aramaki et al. (2007) focused on the physical size of an entity to predict the semantic relation between entities. For example, knowing that a *book* has a physical size of 20 cm × 25 cm and that a *library* has a size of 10 m × 10 m, we can estimate that a library contains a book (content-container relation). Their method acquires knowledge about entity size from the Web (by issuing queries like “*book (\*cm x \*cm)*”), and integrates the knowledge as features for the classification of relations.

Davidov and Rappoport (2010) presented a method for the extraction from the Web and approximation of numerical object attributes such as height and weight. Given an object-attribute pair, the study expands the object into a set of comparable objects and then approximates the numerical values even when no exact value can be found in a text. Aramaki et al. (2007) and Davidov and Rappoport (2010) rely on hand-crafted patterns (e.g., “*Object is \* [unit] tall*”), focusing on a specific set of numerical attributes (e.g., height, weight, size). In contrast, this study can handle any kind of target and situation that is quantified by numbers, e.g., number of people facing a water shortage.

Recently, the RTE community has started to pay some attention to the appropriate processing of numerical expressions. Iftene (2010) presented an approach for matching numerical ranges expressed by a set of phrases (e.g., *more than* and *at least*). Tsuboi et al. (2011) designed hand-crafted rules for matching intervals expressed by temporal expressions. However, these studies do not necessarily focus on semantic processing of numerical expressions; thus, these studies do not normalize units of numerical expressions nor make inferences with numerical common sense.

Sammons et al. (2010) reported that most systems submitted to RTE-5 failed on examples

<sup>1</sup><http://www.itl.nist.gov/iad/mig/tests/ace/ace05/>

<sup>2</sup><http://www.timeml.org/site/timebank/timebank.html>

<sup>3</sup><http://timex2.mitre.org/>

<sup>4</sup><http://timeml.org/site/index.html>

where numeric reasoning was necessary. They argued the importance of aligning numerical quantities and performing numerical reasoning in RTE. LoBue and Yates (2011) identified 20 categories of common-sense knowledge that are prevalent in RTE. One of the categories comprises arithmetic knowledge (including computations, comparisons, and rounding). They concluded that many kinds of the common-sense knowledge have received scarce attention from researchers even though the knowledge is essential to RTE. These studies provided a closer look at the phenomena involved in RTE, but they did not propose a solution for handling numerical expressions.

### 3 Investigation of textual-entailment pairs with numerical expressions

In this section, we investigate textual entailment (TE) pairs in existing corpora in order to study the core phenomena that establish an entailment relation. We used two Japanese TE corpora: RITE (Shima et al., 2011) and Odani et al. (2008). RITE is an evaluation workshop of textual entailment organized by NTCIR-9, and it targets the English, Japanese, and Chinese languages. We used the Japanese portions of the development and training data. Odani et al. (2008) is another Japanese corpus that was manually created. The total numbers of text-hypothesis ( $T$ - $H$ ) pairs are 1,880 (RITE) and 2,471 (Odani).

We manually selected sentence pairs in which one or both of the sentences contained a numerical expression. Here, we define the term *numerical expression* as an expression containing a number or quantity represented by a numeral and a unit. For example, *3 kilometers* is a numerical expression with the numeral *3* and the unit *kilometer*. Note that *intensity of 4* is not a numerical expression because *intensity* is not a unit.

We obtained 371 pairs from the 4,351  $T$ - $H$  pairs. We determined the inferences needed to prove ENTAILMENT or CONTRADICTION of the hypotheses, and classified the 371 pairs into 11 categories. Note that we ignored  $T$ - $H$  pairs in which numerical expressions were unnecessary to prove the entailment relation (e.g., *Socrates was sentenced to death by 500 jury members* and *Socrates was sentenced to death*). Out of 371 pairs, we identified 114 pairs in which numerical expressions played a central role in the entailment relation.

Table 1 summarizes the categories of TE phenomena we found in the data set. The largest category is *numerical matching* (32 pairs). We can infer an entailment relation in this category by aligning two numerical expressions, e.g., *2.2 million*  $\models$  *over 800 thousand*. This is the most fundamental task in numerical reasoning, interpreting the amount (number, unit, and range) in a numerical expression. We address this task in Section 4.1. The second largest category requires common sense about numerical amounts. In order to recognize textual entailment of pairs in this category, we need common-sense knowledge about humans' subjective judgment of numbers. We consider this problem in Section 5.

To summarize, this study covers 37.9% of the instances in Table 1, focusing on the first and second categories. Due to space limitations, we omit the explanations for the other phenomena, which require such things as lexical knowledge, arithmetic operations, and counting. The coverage of this study might seem small, but it is difficult to handle varied phenomena with a unified approach. We believe that this study forms the basis for investigating other phenomena of numerical expressions in the future.

### 4 Collecting numerical expressions from the Web

In this paper, we explore two approaches to acquiring numerical common sense. Both approaches start with extracting numerical expressions and their *context* from the Web. We define a *context* as the verb and its arguments that appear around a numerical expression.

For instance, the context of *3b people* in the sentence *3b people face a water shortage* is “face” and “water shortage.” In order to extract and aggregate numerical expressions in various documents, we converted the numerical expressions into semantic representations (to be described in Section 4.1), and extracted their context (to be described in Section 4.2).

The first approach for acquiring numerical common sense estimates the distribution of numbers that co-occur within a context, and examines whether a given value is large, small, or normal based on that distribution (to be described in Section 5.1). The second approach utilizes textual patterns with which speakers explicitly express their judgment about the value of a numerical ex-

Category	Definition	Example	#
Numerical matching	Aligning numerical expressions in T and H, considering differences in unit, range, etc.	<i>t</i> : It is said that there are about 2.2 million alcoholics in the whole country. <i>h</i> : It is estimated that there are over 800 thousand people who are alcoholics.	32
Numerical common sense	Inferring by interpreting the numerical amount (large or small).	<i>t</i> : In the middle of the 21st century, 7 billion people, corresponding to 70% of the global population, will face a water shortage. <i>h</i> : It is concerning that a serious water shortage will spread around the world in the near future.	12
Lexical knowledge	Inferring by using numerical aspects of word meanings.	<i>t</i> : Mr. and Ms. Sato celebrated their 25th wedding anniversary. <i>h</i> : Mr. and Ms. Sato celebrated their silver wedding anniversary.	12
Arithmetic	Arithmetic operations including addition and subtraction.	<i>t</i> : The number of 2,000-yen bills in circulation has increased to 450 million, in contrast with 440 million 5,000-yen bills. <i>h</i> : The number of 2,000-yen bills in circulation exceeds the number of 5,000-yen bills by 10 million bills.	11
Numeric-range expression of verbs	Numerical ranges expressed by verbs (e.g., <i>exceed</i> ).	<i>t</i> : It is recorded that the maximum wave height reached 13.8 meters during the Sea of Japan Earthquake Tsunami in May 1983. <i>h</i> : During the Sea of Japan Earthquake, the height of the tsunami exceeded 10 meters.	9
Simple Rewrite Rule	This includes various simple rules for rewriting.	<i>t</i> : The strength of Taro’s grip is No. 1 in his class. <i>h</i> : Taro’s grip is the strongest in his class.	7
State change	Expressing the change of a value by a multiplier or ratio.	<i>t</i> : Consumption of pickled plums is 1.5 times the rate of 20 years ago. <i>h</i> : Consumption of pickled plums has increased.	6
Ordinal numbers	Inference by interpreting ordinal numbers.	<i>t</i> : Many precious lives were sacrificed in the Third World War. <i>h</i> : So far, there have been at least three World Wars.	6
Temporal expression	Inference by interpreting temporal expressions such as anniversary, age, and ordinal numbers.	<i>t</i> : Mr. and Ms. Sato celebrate their 25th wedding anniversary. <i>h</i> : Mr. and Ms. Sato got married 25 years ago.	3
Count	Counting up the number of various entities.	<i>t</i> : In Japan, there are the Asian Triopsidae, the American Triopsidae, and the European Triopsidae. <i>h</i> : In Japan, there are 3 types of Triopsidae.	3
Others			15
All			116

Table 1: Frequency and simple definitions for each category of the entailment phenomena in the survey.

Numerical Expression	Semantic representation		
	Value	Unit	Mod.
<i>about seven grams</i>	7	g	about
<i>roughly 7 kg</i>	7000	g	about
<i>as heavy as 7 tons</i>	$7 \times 10^6$	g	large
<i>as cheap as \$1</i>	1	\$	small
<i>30–40 people</i>	[30, 40]	nin ( <i>people</i> )	
<i>more than 30 cars</i>	30	dai ( <i>cars</i> )	over
<i>7 km per hour</i>	7000	m/h	

Table 2: Normalized representation examples

String	Operation
gram(s)	set-unit: ‘g’
kilogram(s)	set-unit: ‘g’; multiply-value: 1,000
kg	set-unit: ‘g’; multiply-value: 1,000
ton(s)	set-unit: ‘g’; multiply-value: 1,000,000
nin ( <i>people</i> )	set-unit: ‘nin’ ( <i>person</i> )
about	set-modifier: ‘about’
as many as	set-modifier: ‘large’
as little as	set-modifier: ‘small’

Table 3: An example of unit/modifier dictionary

pression (to be explained in Section 5.2).

In this study, we acquired numerical common sense from a collection of 8 billion sentences in 100 million Japanese Web pages (Shinzato et al., 2012). For this reason, we originally designed text patterns specialized for Japanese dependency trees. For the sake of the readers’ understanding, this paper uses examples with English translations for explaining language-independent concepts, and both Japanese and English translations for explaining language-dependent concepts.

#### 4.1 Extracting and normalizing numerical expressions

The first step for collecting numerical expressions is to recognize when a numerical expression is mentioned and then to normalize it into a semantic representation. This is the most fundamental

step in numerical reasoning and has a number of applications. For example, this step handles cases of *numerical matching*, as in Table 1.

The semantic representation of a numerical expression consists of three fields: the value or range of the real number(s)<sup>5</sup>, the unit (a string), and the optional modifiers. Table 2 shows some examples of numerical expressions and their semantic representations. During normalization, we identified spelling variants (e.g., *kilometer* and *km*) and transformed auxiliary units into their corresponding canonical units (e.g., *2 tons* and *2,000 kg* to *2,000,000 grams*). When a numerical expression is accompanied by a modifier such as *over*, *about*, or *more than*, we updated the value and modifier fields appropriately.

<sup>5</sup>Internally, all values are represented by ranges (e.g., 75 is represented by the range [75, 75]).

We developed an extractor and a normalizer for Japanese numerical expressions<sup>6</sup>. We will outline the algorithm used in the normalizer with an example sentence: “Roughly three thousand kilograms of meats have been provided every day.”

1. Find numbers in the text by using regular expressions and convert the non-Arabic numbers into their corresponding Arabic numbers. For example, we find *three thousand*<sup>7</sup> and represent it as 3,000.
2. Check whether the words that precede or follow the number are units that are registered in the dictionary. Transform any auxiliary units. In the example, we find that *kilograms*<sup>8</sup> is a unit. We multiply the value 3,000 by 1,000, and obtain the value 3,000,000 with the unit *g*.
3. Check whether the words that precede or follow the number have a modifier that is registered in the dictionary. Update the value and modifier fields if necessary. In the example, we find *roughly* and set *about* in the modifier field.

We used a dictionary<sup>9</sup> to perform procedures 2 and 3 (Table 3). If the words that precede or follow an extracted number match an entry in the dictionary, we change the semantic representation as described in the operation.

The modifiers ‘large’ and ‘small’ require elaboration because the method in Section 5.2 relies heavily on these modifiers. We activated the modifier ‘large’ when a numerical expression occurred with the Japanese word *mo*, which roughly corresponds to *as many as*, *as large as*, or *as heavy as* in English<sup>10</sup>. Similarly, we activated the modifier ‘small’ when a numerical expression occurred with the word *shika*, which roughly corresponds to *as little as*, *as small as*, or *as light as*<sup>11</sup>. These modifiers are important for this study, reflecting the writer’s judgment about the amount.

<sup>6</sup>The software is available at <http://www.cl.ecei.tohoku.ac.jp/~katsuma/software/normalizeNumexp/>

<sup>7</sup>In Japanese 3,000 is denoted by the Chinese symbols “三千”.

<sup>8</sup>We write kilograms as “キログラム” in Japanese.

<sup>9</sup>The dictionary is bundled with the tool. See Footnote 6.

<sup>10</sup>In Japanese, we can use the word *mo* with a numerical expression to state that the amount is ‘large’ regardless of how large it is (e.g., large, big, many, heavy).

<sup>11</sup>Similarly, we can use the word *shika* with any adjective.

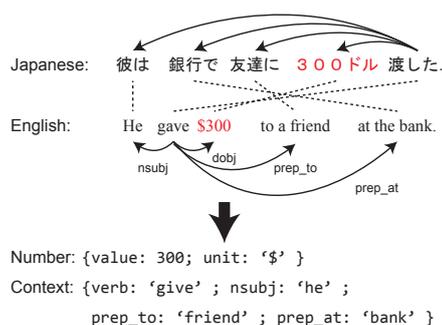


Figure 1: Example of context extraction

## 4.2 Extraction of context

The next step in acquiring numerical common sense is to capture the context of numerical expressions. Later, we will aggregate numbers that share the same context (see Section 5). The context of a numerical expression should provide sufficient information to determine what it measures. For example, given the sentence, “He gave \$300 to a friend at the bank,” it would be better if we could generalize the context to *someone gives money to a friend* for the numerical expression \$300. However, it is a nontrivial task to design an appropriate representation of varying contexts. For this reason, we employ a simple rule to capture the context of numerical expressions: we represent the context with the verb that governs the numerical expression and its typed arguments.

Figure 1 illustrates the procedure for extracting the context of a numerical expression<sup>12</sup>. The component in Section 4.1 recognizes \$300 as a numerical expression, then normalizes it into a semantic representation. Because the numerical expression is a dependent of the verb *gave*, we extract the verb and its arguments (except for the numerical expression itself) as the context. After removing inflections and function words from the arguments, we obtain the context representation of Figure 1.

## 5 Acquiring numerical common sense

In this section, we present two approaches for acquiring numerical common sense from a collection of numerical expressions and their contexts. Both approaches start with collecting the numbers (in semantic representation) and contexts of numerical expressions from a large number of sentences (Shinzato et al., 2012), and storing them

<sup>12</sup>The English dependency tree might look peculiar because it is translated from the Japanese dependency tree.

in a database. When a context and a value are given for a prediction (hereinafter called the query context and query value, respectively), these approaches judge whether the query value is large, small, or normal.

### 5.1 Distribution-based approach

Given a query context and query value, this approach retrieves numbers associated with the query context and draws a distribution of normalized numbers. This approach considers the distribution estimated for the query context and determines if the value is within the top 5 percent (large), within the bottom 5 percent (small), or is located in between these regions (normal).

The underlying assumption of this approach is that the real distribution of a query (e.g., *money given to a friend*) can be approximated by the distribution of numbers co-occurring with the context (e.g., *give* and *friend*) on the Web. However, the context space generated in Section 4.2 may be too sparse to find numbers in the database, especially when a query context is fine-grained. Therefore, when no item is retrieved for the query context, we employ a backoff strategy to drop some of the uninformative elements in the query context: elements are dropped from the context based on the type of argument, in this order: *he* (prep\_to), *kara* (prep\_from), *ha* (nsubj), *yoru* (prep\_from), *made* (prep\_to), *nite* (prep\_at), *de* (prep\_at, prep\_by), *ni* (prep\_at), *wo* (doobj), *ga* (nsubj), and verb.

### 5.2 Clue-based approach

This approach utilizes textual clues with which a speaker explicitly expresses his or her judgment about the amount of a numerical expression. We utilize large and small modifiers (described in Section 4.1), which correspond to textual clues *mo* (*as many as*, *as large as*) and *shika* (*only*, *as few as*), respectively, for detecting humans' judgments. For example, we can guess that \$300 is large if we find an evidential sentence<sup>13</sup>, *He gave as much as \$100 to a friend*.

Similarly to the distribution-based approach, this approach retrieves numbers associated with the query context. This approach computes the

<sup>13</sup>Although the sentence states a judgment about \$100, we can infer that \$300 is also large because \$300 > \$100.

largeness  $L(x)$  of a value  $x$ :

$$L(x) = \frac{p_l(x)}{p_s(x) + p_l(x)}, \quad (1)$$

$$p_l(x) = \frac{|\{r|r_v < x \wedge r_m \ni large\}|}{|\{r|r_m \ni large\}|}, \quad (2)$$

$$p_s(x) = \frac{|\{r|r_v > x \wedge r_m \ni small\}|}{|\{r|r_m \ni small\}|}. \quad (3)$$

In these equations,  $r$  denotes a retrieved item for the query context, and  $r_v$  and  $r_m$  represent the normalized value and modifier flags, respectively, of the item  $r$ . The numerator of Equation 2 counts the number of numerical expressions that support the judgment that  $x$  is large<sup>14</sup>, and its denominator counts the total number of numerical expressions with *large* as a modifier. Therefore,  $p_l(x)$  computes the ratio of times there is textual evidence that says that  $x$  is large, to the total number of times there is evidences with *large* as a modifier. In an analogous way,  $p_s(x)$  is defined to be the ratio for evidence that says  $x$  is small. Hence,  $L(x)$  approaches 1 if everyone on the Web claims that  $x$  is large, and approaches 0 if everyone claims that  $x$  is small. This approach predicts *large* if  $L(x) > 0.95$ , *small* if  $L(x) < 0.05$ , and *normal* otherwise.

## 6 Experiments

### 6.1 Normalizing numerical expressions

We evaluated the method that we described in Section 4.1 for extracting and normalizing numerical expressions. In order to prepare a gold-standard data set, we obtained 1,041 sentences by randomly sampling about 1% of the sentences containing numbers (Arabic digits and/or Chinese numerical characters) in a Japanese Web corpus (100 million pages) (Shinzato et al., 2012). For every numerical expression in these sentences, we manually determined a tuple of the normalized value, unit, and modifier. Here, non-numerical expressions such as temporal expressions, telephone numbers, and postal addresses, which were very common, were beyond the scope of the project<sup>15</sup>. We obtained 329 numerical expressions from the 1,041 sentences.

We evaluated the correctness of the extraction and normalization by measuring the precision and

<sup>14</sup>This corresponds to the events where we find an evidence expression "as many as  $r_v$ ", where  $r_v < x$ .

<sup>15</sup>If a tuple was extracted from a non-numerical expression, we regarded this as a false positive

recall using the gold-standard data set<sup>16</sup>. Our method performed with a precision of 0.78 and a recall of 0.92. Most of the false negatives were caused by the incompleteness of the unit dictionary. For example, the proposed method could not identify *1Ghz* as a numerical expression because the unit dictionary did not register *Ghz* but *GHz*. It is trivial to improve the recall of the method by enriching the unit dictionary.

The major cause of false positives was the semantic ambiguity of expressions. For example, the proposed method identified *Seven Hills* as a numerical expression although it denotes a location name. In order to reduce false positives, it may be necessary to utilize broader contexts when locating numerical expressions; this could be done by using, for example, a named entity recognizer. This is the next step to pursue in future work.

However, these errors do not have a large effect on the estimation of the distribution of the numerical values that occur with specific named entities and idiomatic phrases. Moreover, as explained in Section 5, we draw distributions for fine-grained contexts of numerical expressions. For these reasons, we think that the current performance is sufficient for acquiring numerical common sense.

## 6.2 Acquisition of numerical common sense

### 6.2.1 Preparing an evaluation set

We built a gold-standard data set for numerical common sense. We applied the method in Section 4.1 to sentences sampled at random from the Japanese Web corpus (Shinzato et al., 2012), and we extracted 2,000 numerical expressions. We asked three human judges to annotate every numerical expression with one of six labels, *small*, *relatively small*, *normal*, *relatively large*, *large*, and *unsure*. The label *relatively small* could be applied to a numerical expression when the judge felt that the amount was rather small (below the normal) but hesitated to label it *small*. The label *relatively large* was defined analogously. We gave the following criteria for labeling an item as *unsure*: when the judgment was highly dependent on the context; when the sentence was incomprehensible; and when it was a non-numerical expressions (false positives of the method are discussed in Section 4.1).

Table 4 reports the inter-annotator agreement.

<sup>16</sup>All fields (value, unit, modifier) of the extracted tuple must match the gold-standard data set.

Agreement	# expressions
3 annotators	735 (36.7%)
2 annotators	963 (48.2%)
no agreement	302 (15.1%)
Total	2000 (100.0%)

Table 4: Inter-annotator agreement

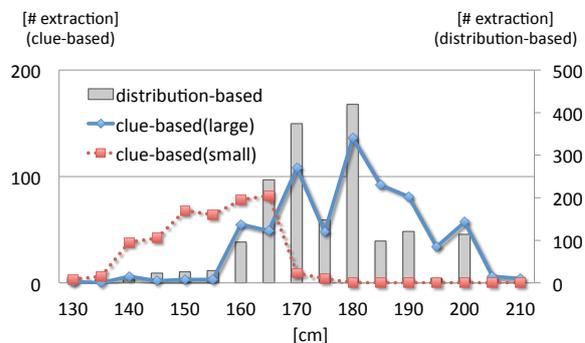


Figure 2: Distributions of numbers with large and small modifiers for the context *human's height*.

For the evaluation of numerical expressions in the data set, we used those for which at least two annotators assigned the same label. After removing the *unsure* instances, we obtained 640 numerical expressions (20 *small*, 35 *relatively small*, 152 *normal*, 263 *relatively large*, and 170 *large*) as the evaluation set.

### 6.2.2 Results

The proposed method extracted about 23 million pairs of numerical expressions and their context from the corpus (with 100 million Web pages). About 15% of the extracted pairs were accompanied by either a large or small modifier. Figure 2 depicts the distributions of the context *human's height* produced by the distribution-based and clue-based approaches. These distributions are quite reasonable as common-sense knowledge: we can interpret that numbers under 150 cm are perceived as small and those above 180 cm as large.

We measured the correctness of the proposed methods on the gold-standard data. For this evaluation, we employed two criteria for correctness: strict and lenient. With the strict criterion, the method must predict a label identical to that in the gold-standard. With the lenient criterion, the method was also allowed to predict either *large/small* or *normal* when the gold-standard label was *relatively large/small*.

Table 5 reports the precision (P), recall (R), F1 (F1), and accuracy (Acc) of the proposed methods.

No.	System	Gold	Sentence	Remark
1	small	small	I think that <u>three men</u> can create such a great thing in the world.	Correct
2	normal	normal	I have <u>two cats</u> .	Correct
3	large	large	It's <u>above 32 centigrade</u> .	Correct
4	large	large	I earned <u>10 million yen</u> from horse racing.	Correct
5	small	normal	There are <u>2 reasons</u> .	<i>Difficulty in judging small.</i> Since a few people say, "There are only 2 reasons," our approach predicted a <i>small</i> label.
6	small	large	Ten or more people came, and my eight-mat room was packed.	<i>Difficulty in modeling the context</i> because this sentence omits the locational argument for the verb <i>came</i> . We should extract the context as <i>the number of people who came to my eight-mat room</i> instead of <i>the number of people who came</i> .
7	small	normal	I have <u>two friends</u> who have broken up with their boyfriends recently.	<i>Difficulty in modeling the context.</i> We should extract context as <i>the number of friends who have broken up with their boyfriends recently</i> instead of <i>the number of friends</i> .
8	small	large		<i>Lack of knowledge.</i> We extract the context as <i>the number of heads of a turtle</i> , but no corresponding information was found on the Web.

Table 6: Output example and error analysis. We present translations of the sentences, which were originally in Japanese.

Approach	Label	P	R	F1	Acc
Distribution	large+	0.892	0.498	0.695	0.760
	normal+	0.753	0.935	0.844	
	small+	0.273	0.250	0.262	
Distribution	large	0.861	0.365	0.613	0.590
	normal	0.529	0.908	0.719	
	small	0.222	0.100	0.161	
Clue	large+	0.923	0.778	0.851	0.770
	normal+	0.814	0.765	0.790	
	small+	0.228	0.700	0.464	
Clue	large	0.896	0.659	0.778	0.620
	normal	0.593	0.586	0.590	
	small	0.164	0.550	0.357	

Table 5: Precision (P), recall (R), F1 score (F1), and accuracy (Acc) of the acquisition of numerical common sense.

Labels with the suffix ‘+’ correspond to the lenient criterion. The clue-based approach achieved 0.851 F1 (for large), 0.790 F1 (for normal), and 0.464 (for small) with the lenient criterion. The performance is surprisingly good, considering the subjective nature of this task.

The clue-based approach was slightly better than the distribution-based approach. In particular, the clue-based approach is good at predicting large and small labels, whereas the distribution-based approach is good at predicting normal labels. We found some targets for which the distribution on the Web is skewed from the ‘real’ distribution. For example, let us consider the distribution of the context “*the amount of money that a person wins in a lottery*”. We can find a number of sentences like *if you won the 10-million-dollar lottery, ...*. In other words, people talk about a large amount of money even if they did not win any money at all. In order to remedy this problem,

we may need to enrich the context representation by introducing, for example, the factuality of an event.

### 6.2.3 Discussion

Table 6 shows some examples of predictions from the clue-based approach. Because of space limitations, we mention only the false instances of this approach.

The clue-based approach tends to predict *small* even if the gold-standard label is *normal*. About half of the errors of the clue-based approach were of this type; this is why the precision for *small* and the recall for *normal* are low. The cause of this error is exemplified by the sentence, “there are two reasons.” Human judges label *normal* to the numerical expression *two reasons*, but the method predicts *small*. This is because a few people say *there are only two reasons*, but no one says *there are as many as two reasons*. In order to handle these cases, we may need to incorporate the distribution information with the clue-based approach.

We found a number of examples for which modeling the context is difficult. Our approach represents the context of a numerical expression with the verb that governs the numerical expression and its typed arguments. However, this approach sometimes misses important information, especially when an argument of the verb is omitted (Example 6). The approach also suffers from the relative clause in Example 7, which conveys an essential context of the number. These are similar to the scope-ambiguity problem such as encoun-

tered with negation and quantification; it is difficult to model the scope when a numerical expression refers to a situation.

Furthermore, we encountered some false examples even when we were able to precisely model the context. In Example 8, the proposed method was unable to predict the label correctly because no corresponding information was found on the Web. The proposed method might more easily predict a label if we could generalize the word *turtle* as *animal*. It may be worth considering using language resources (e.g., WordNet) to generalize the context.

## 7 Conclusions

We proposed novel approaches for acquiring numerical common sense from a collection of texts. The approaches collect numerical expressions and their contexts from the Web, and acquire numerical common sense by considering the distributions of normalized numbers and textual clues such as *mo (as many as) and shika (only, as few as)*. The experimental results showed that our approaches can successfully judge whether a given amount is large, small, or normal. The implementations and data sets used in this study are available on the Web<sup>17</sup>. We believe that acquisition of numerical common sense is an important step towards a deeper understanding of inferences with numbers.

There are three important future directions for this research. One is to explore a more sophisticated approach for precisely modeling the contexts of numbers. Because we confirmed in this paper that these two approaches have different characteristics, it would be interesting to incorporate textual clues into the distribution-based approach by using, for example, machine learning techniques. Finally, we are planning to address the ‘third phase’ of the example explained in Section 1: associating *many people face a water shortage with a serious water shortage*.

## Acknowledgments

This research was partly supported by JST, PRESTO. This research was partly supported by JSPS KAKENHI Grant Numbers 23240018 and 23700159.

<sup>17</sup><http://www.cl.ecei.tohoku.ac.jp/~katsuma/resource/numerical.common.sense/>

## References

- Eiji Aramaki, Takeshi Imai, Kengo Miyo, and Kazuhiko Ohe. 2007. Uth: Svm-based semantic relation classification using physical works. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 464–467.
- Anton Bakalov, Ariel Fuxman, Partha Pratim Talukdar, and Soumen Chakrabarti. 2011. SCAD: collective discovery of attribute values. In *Proceedings of the 20th international conference on World wide web, WWW '11*, pages 447–456.
- Somnath Banerjee, Soumen Chakrabarti, and Ganesh Ramakrishnan. 2009. Learning to rank for quantity consensus queries. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '09*, pages 243–250.
- Luisa Bentivogli, Elena Cabrio, Ido Dagan, Danilo Giampiccolo, Medea Lo Leggio, and Bernardo Magnini. 2010. Building textual entailment specialized data sets: a methodology for isolating linguistic phenomena relevant to inference. *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 3542–3549.
- Elena Cabrio and Bernardo Magnini. 2011. Towards component-based textual entailment. In *Proceedings of the Ninth International Conference on Computational Semantics, IWCS '11*, pages 320–324.
- Jennifer Chu-Carroll, David A. Ferrucci, John M. Prager, and Christopher A. Welty. 2003. Hybridization in question answering systems. In *New Directions in Question Answering '03*, pages 116–121.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190.
- Dmitry Davidov and Ari Rappoport. 2010. Extraction and approximation of numerical attributes from the web. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1308–1317.
- Marcus Fontoura, Ronny Lempel, Runping Qi, and Jason Zien. 2006. Inverted index support for numeric search. *Internet Mathematics*, 3(2):153–185.
- Adrian Iftene and Mihai-Alex Moruz. 2010. UAIC participation at RTE-6. In *Proceedings of the Third Text Analysis Conference (TAC 2010) November*.
- Douglas B Lenat. 1995. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.
- Peter LoBue and Alexander. Yates. 2011. Types of common-sense knowledge needed for recognizing

- textual entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 329–334.
- Véronique Moriceau. 2006. Generating intelligent numerical answers in a question-answering system. In *Proceedings of the Fourth International Natural Language Generation Conference*, INLG '06, pages 103–110.
- Michitaka Odani, Tomohide Shibata, Sadao Kurohashi, and Takayuki Nakata. 2008. Building data of japanese text entailment and recognition of inferencing relation based on automatic achieved similar expression. In *Proceeding of 14th Annual Meeting of the Association for "atural Language Processing*, pages 1140–1143.
- John M. Prager, Jennifer Chu-Carroll, Krzysztof Czuba, Christopher A. Welty, Abraham Ittycheriah, and Ruchi Mahindru. 2003. IBM's PIQUANT in TREC2003. In *TREC*, pages 283–292.
- Mark Sammons, Vinod V.G. Vydiswaran, and Dan Roth. 2010. Ask not what textual entailment can do for you... In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1199–1208.
- Hideki Shima, Hiroshi Kanayama, Cheng-Wei Lee, Chuan-Jie Lin, Teruko Mitamura, Yusuke Miyao, Shuming Shi, and Koichi Takeda. 2011. Overview of ntcir-9 rite: Recognizing inference in text. In *Proceeding of NTCIR-9 Workshop Meeting*, pages 291–301.
- Keiji Shinzato, Tomohide Shibata, Daisuke Kawahara, and Sadao Kurohashi. 2012. Tsubaki: An open search engine infrastructure for developing information access methodology. *Journal of Information Processing*, 20(1):216–227.
- Assaf Toledo, Sophia Katrenko, Stavroula Alexandropoulou, Heidi Klockmann, Asher Stern, Ido Dagan, and Yoad Winter. 2012. Semantic annotation for textual entailment recognition. In *Proceedings of the 11th Mexican International Conference on Artificial Intelligence*, MICAI '12.
- Yuta Tsuboi, Hiroshi Kanayama, Masaki Ohno, and Yuya Unno. 2011. Syntactic difference based approach for ntcir-9 rite task. In *Proceedings of the 9th NTCIR Workshop*, pages 404–411.
- Minoru Yoshida, Issei Sato, Hiroshi Nakagawa, and Akira Terada. 2010. Mining numbers in text using suffix arrays and clustering based on dirichlet process mixture models. *Advances in Knowledge Discovery and Data Mining*, pages 230–237.

# Probabilistic Domain Modelling With Contextualized Distributional Semantic Vectors

**Jackie Chi Kit Cheung**

University of Toronto  
10 King's College Rd., Room 3302  
Toronto, ON, Canada M5S 3G4  
jcheung@cs.toronto.edu

**Gerald Penn**

University of Toronto  
10 King's College Rd., Room 3302  
Toronto, ON, Canada M5S 3G4  
gpenn@cs.toronto.edu

## Abstract

Generative probabilistic models have been used for content modelling and template induction, and are typically trained on small corpora in the target domain. In contrast, vector space models of distributional semantics are trained on large corpora, but are typically applied to domain-general lexical disambiguation tasks. We introduce Distributional Semantic Hidden Markov Models, a novel variant of a hidden Markov model that integrates these two approaches by incorporating contextualized distributional semantic vectors into a generative model as observed emissions. Experiments in slot induction show that our approach yields improvements in learning coherent entity clusters in a domain. In a subsequent extrinsic evaluation, we show that these improvements are also reflected in multi-document summarization.

## 1 Introduction

Detailed domain knowledge is crucial to many NLP tasks, either as an input for language understanding, or as the goal itself, to acquire such knowledge. For example, in information extraction, a list of slots in the target domain is given to the system, and in natural language generation, content models are trained to learn the content structure of texts in the target domain for information structuring and automatic summarization.

Generative probabilistic models have been one popular approach to content modelling. An important advantage of this approach is that the structure of the model can be adapted to fit the assumptions about the structure of the domain and the nature of the end task. As this field has progressed, the formal structures that are assumed to represent a

domain have increased in complexity and become more hierarchical. Earlier work assumes a flat set of topics (Barzilay and Lee, 2004), which are expressed as states of a latent random variable in the model. Later work organizes topics into a hierarchy from general to specific (Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tur, 2010). Recently, Cheung et al. (2013) formalized a domain as a set of *frames* consisting of prototypical sequences of *events*, *slots*, and *slot fillers* or *entities*, inspired by classical AI work such as Schank and Abelson's (1977) scripts. We adopt much of this terminology in this work. For example, in the CRIMINAL INVESTIGATIONS domain, there may be events such as a murder, an investigation of the crime, an arrest, and a trial. These would be indicated by *event heads* such as *kill*, *arrest*, *charge*, *plead*. Relevant slots would include VICTIM, SUSPECT, AUTHORITIES, PLEA, etc.

One problem faced by this line of work is that, by their nature, these models are typically trained on a small corpus from the target domain, on the order of hundreds of documents. The small size of the training corpus makes it difficult to estimate reliable statistics, especially for more powerful features such as higher-order N-gram features or syntactic features.

By contrast, distributional semantic models are trained on large, domain-general corpora. These methods model word meaning using the contexts in the training corpus in which the word appears. The most popular approach today is a vector space representation, in which each dimension corresponds to some context word, and the value at that dimension corresponds to the strength of the association between the context word and the target word being modelled. A notion of word similarity arises naturally from these models by comparing the similarity of the word vectors, for example by using a cosine measure. Recently, these models have been extended by considering how distribu-

tional representations can be modified depending on the specific context in which the word appears (Mitchell and Lapata, 2008, for example). Contextualization has been found to improve performance in tasks like lexical substitution and word sense disambiguation (Thater et al., 2011).

In this paper, we propose to inject contextualized distributional semantic vectors into generative probabilistic models, in order to combine their complementary strengths for domain modelling. There are a number of potential advantages that distributional semantic models offer. First, they provide domain-general representations of word meaning that cannot be reliably estimated from the small target-domain corpora on which probabilistic models are trained. Second, the contextualization process allows the semantic vectors to implicitly encode disambiguated word sense and syntactic information, without further adding to the complexity of the generative model.

Our model, the Distributional Semantic Hidden Markov Model (DSHMM), incorporates contextualized distributional semantic vectors into a generative probabilistic model as observed emissions. We demonstrate the effectiveness of our model in two domain modelling tasks. First, we apply it to slot induction on guided summarization data over five different domains. We show that our model outperforms a baseline version of our method that does not use distributional semantic vectors, as well as a recent state-of-the-art template induction method. Then, we perform an extrinsic evaluation using multi-document summarization, wherein we show that our model is able to learn event and slot topics that are appropriate to include in a summary. From a modelling perspective, these results show that probabilistic models for content modelling and template induction benefit from distributional semantics trained on a much larger corpus. From the perspective of distributional semantics, this work broadens the variety of problems to which distributional semantics can be applied, and proposes methods to perform inference in a probabilistic setting beyond geometric measures such as cosine similarity.

## 2 Related Work

Probabilistic content models were proposed by Barzilay and Lee (2004), and related models have since become popular for summarization (Fung and Ngai, 2006; Haghghi and Vanderwende,

2009), and information ordering (Elsner et al., 2007; Louis and Nenkova, 2012). Other related generative models include topic models and structured versions thereof (Blei et al., 2003; Gruber et al., 2007; Wallach, 2008). In terms of domain learning in the form of template induction, heuristic methods involving multiple clustering steps have been proposed (Filatova et al., 2006; Chambers and Jurafsky, 2011). Most recently, Cheung et al. (2013) propose PROFINDER, a probabilistic model for frame induction inspired by content models. Our work is similar in that we assume much of the same structure within a domain and consequently in the model as well (Section 3), but whereas PROFINDER focuses on finding the “correct” number of frames, events, and slots with a nonparametric method, this work focuses on integrating global knowledge in the form of distributional semantics into a probabilistic model. We adopt one of their evaluation procedures and use it to compare with PROFINDER in Section 5.

Vector space models form the basis of modern information retrieval (Salton et al., 1975), but only recently have distributional models been proposed that are compositional (Mitchell and Lapata, 2008; Clark et al., 2008; Grefenstette and Sadrzadeh, 2011, *inter alia*), or that contextualize the meaning of a word using other words in the same phrase (*co-compositionality*) (Erk and Padó, 2008; Dinu and Lapata, 2010; Thater et al., 2011). We recently showed how such models can be evaluated for their ability to support semantic inference for use in complex NLP tasks like question answering or automatic summarization (Cheung and Penn, 2012).

Combining distributional information and probabilistic models has actually been explored in previous work. Usually, an ad-hoc clustering step precedes training and is used to bias the initialization of the probabilistic model (Barzilay and Lee, 2004; Louis and Nenkova, 2012), or the clustering is interleaved with iterations of training (Fung et al., 2003). By contrast, our method better modularizes the two, and provides a principled way to train the model. More importantly, previous ad-hoc clustering methods only use distributional information derived from the target domain itself; initializing based on domain-general distributional information can be problematic because it can bias training towards a local optimum that is inappropriate for the target domain, leading to poor per-

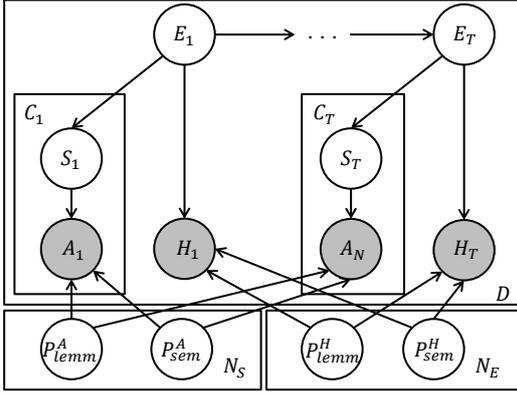


Figure 1: Graphical representation of our model. Distributions that generate the latent variables and hyperparameters are omitted for clarity.

formance.

### 3 Distributional Semantic Hidden Markov Models

We now describe the DSHMM model. This model can be thought of as an HMM with two layers of latent variables, representing events and slots in the domain. Given a document consisting of a sequence of  $T$  clauses headed by propositional heads  $\vec{H}$  (verbs or event nouns), and argument noun phrases  $\vec{A}$ , a DSHMM models the joint probability of observations  $\vec{H}$ ,  $\vec{A}$ , and latent random variables  $\vec{E}$  and  $\vec{S}$  representing domain events and slots respectively; i.e.,  $P(\vec{H}, \vec{A}, \vec{E}, \vec{S})$ .

The basic structure of our model is similar to PROFINDER. Each timestep in the model generates one clause in the document. More specifically, it generates the event heads and arguments which are crucial in identifying events and slots. We assume that event heads are verbs or event nouns, while arguments are the head words of their syntactically dependent noun phrases. We also assume that the sequence of clauses and the clause-internal syntactic structure are fixed, for example by applying a dependency parser. Within each clause, a hierarchy of latent and observed variables maps to corresponding elements in the clause (Table 1), as follows:

**Event Variables** At the top-level, a categorical latent variable  $E_t$  with  $N_E$  possible states represents the event that is described by clause  $t$ . Its value is conditioned on the previous time step’s event variable, following the standard, first-order Markov assumption ( $P^E(E_t|E_{t-1})$ , or  $P_{init}^E(E_1)$

Node	Component	Textual unit
$E_t$	Event	Clause
$S_{ta}$	Slot	Noun phrase
$H_t$	Event head	Verb/event noun
$A_{ta}$	Event argument	Noun phrase

Table 1: The correspondence between nodes in our graphical model, the domain components that they model, and the related elements in the clause.

for the first clause). The internal structure of the clause is generated by conditioning on the state of  $E_t$ , including the head of the clause, and the slots for each argument in the clause.

**Slot Variables** Categorical latent variables with  $N_S$  possible states represent the slot that an argument fills, and are conditioned on the event variable in the clause,  $E_t$  (i.e.,  $P^S(S_{ta}|E_t)$ , for the  $a$ th slot variable). The state of  $S_{ta}$  is then used to generate an argument  $A_{ta}$ .

**Head and Argument Emissions** The head of the clause  $H_t$  is conditionally dependent on  $E_t$ , and each argument  $A_{ta}$  is likewise conditioned on its slot variable  $S_{ta}$ . Unlike in most applications of HMMs in text processing, in which the representation of a token is simply its word or lemma identity, tokens in DSHMM are also associated with a vector representation of their meaning *in context* according to a distributional semantic model (Section 3.1). Thus, the emissions can be decomposed into pairs  $H_t = (\text{lemma}(H_t), \text{sem}(H_t))$  and  $A_{ta} = (\text{lemma}(A_{ta}), \text{sem}(A_{ta}))$ , where *lemma* and *sem* are functions that return the lemma identity and the semantic vector respectively. The probability of the head of a clause is thus:

$$P^H(H_t|E_t) = P_{lemm}^H(\text{lemma}(H_t)|E_t) \quad (1) \\ \times P_{sem}^H(\text{sem}(H_t)|E_t),$$

and the probability of a clausal argument is likewise:

$$P^A(A_{ta}|S_{ta}) = P_{lemm}^A(\text{lemma}(A_{ta})|S_{ta}) \quad (2) \\ \times P_{sem}^A(\text{sem}(A_{ta})|S_{ta}).$$

All categorical distributions are smoothed using add- $\delta$  smoothing (i.e., uniform Dirichlet priors). Based on the independence assumptions described above, the joint probability distribution can be fac-

tored into:

$$\begin{aligned}
P(\vec{H}, \vec{A}, \vec{E}, \vec{S}) &= P_{init}^E(E_1) \\
&\times \prod_{t=2}^T P^E(E_t|E_{t-1}) \prod_{t=1}^T P^H(H_t|E_t) \\
&\times \prod_{t=1}^T \prod_{a=1}^{C_t} P^S(S_{ta}|E_t) P^A(A_{ta}|S_{ta}).
\end{aligned} \quad (3)$$

### 3.1 Vector Space Models of Semantics

In this section, we describe several methods for producing the semantic vectors associated with each event head or argument; i.e., the function *sem*. We chose several simple, but widely studied models, to investigate whether they can be effectively integrated into DSHMM. We start with a description of the training of a basic model without any contextualization, then describe several contextualized models based on recent work.

**Simple Vector Space Model** In the basic version of the model (SIMPLE), we train a term-context matrix, where rows correspond to target words, and columns correspond to context words. Training begins by counting context words that appear within five words of the target word, ignoring stopwords. We then convert the raw counts to positive pointwise mutual information scores, which has been shown to improve word similarity correlation results (Turney and Pantel, 2010). We set thresholds on the frequencies of words for inclusion as target and context words (given in Section 4). Target words which fall below the threshold are modelled as *UNK*. All the methods below start from this basic vector representation.

**Component-wise Operators** Mitchell and Lapata (2008) investigate using component-wise operators to combine the vectors of verbs and their intransitive subjects. We use component-wise operators to contextualize our vectors, but by combining with all of the arguments, and regardless of the event head’s category. Let event head  $h$  be the syntactic head of a number of arguments  $a_1, a_2, \dots, a_m$ , and  $\vec{v}_h, \vec{v}_{a_1}, \vec{v}_{a_2}, \dots, \vec{v}_{a_m}$  be their respective vector representations according to the SIMPLE method. Then, their contextualized vectors  $\vec{c}_h^{M\&L}, \vec{c}_{a_1}^{M\&L}, \dots, \vec{c}_{a_m}^{M\&L}$  would be:

$$\vec{c}_h^{M\&L} = \vec{v}_h \odot \left( \bigodot_{i=1}^m \vec{v}_{a_i} \right) \quad (4)$$

$$\vec{c}_{a_i}^{M\&L} = \vec{v}_{a_i} \odot \vec{v}_h, \forall i = 1 \dots m, \quad (5)$$

where  $\odot$  represents a component-wise operator, addition or multiplication, and  $\bigodot$  represents its repeated application. We tested component-wise addition (M&L+) and multiplication (M&L $\times$ ).

**Selectional Preferences** Erk and Padó (2008) (E&P) incorporate *inverse selectional preferences* into their contextualization function. The intuition is that a word should be contextualized such that its vector representation becomes more similar to the vectors of other words that its dependency neighbours often take in the same syntactic position. For example, suppose *catch* is the head of the noun *ball*, in the relation of a direct object. Then, the vector for *ball* would be contextualized to become similar to the vectors for other frequent direct objects of *catch*, such as *baseball*, or *cold*. Likewise, the vector for *catch* would be contextualized to become similar to the vectors for *throw*, *hit*, etc. Formally, let  $h$  take  $a$  as its argument in relation  $r$ . Then:

$$\vec{c}_h^{E\&P} = \vec{v}_h \times \prod_{i=1}^m \sum_{w \in \mathcal{L}} freq(w, r, a_i) \cdot \vec{v}_w, \quad (6)$$

$$\vec{c}_a^{E\&P} = \vec{v}_a \times \sum_{w \in \mathcal{L}} freq(h, r, w) \cdot \vec{v}_w, \quad (7)$$

where  $freq(h, r, a)$  is the frequency of  $h$  occurring as the head of  $a$  in relation  $r$  in the training corpus,  $\mathcal{L}$  is the lexicon, and  $\times$  represents component-wise multiplication.

### Dimensionality Reduction and Vector Emission

After contextualization, we apply singular value decomposition (SVD) for dimensionality reduction to reduce the number of model parameters, keeping the  $k$  most significant singular values and vectors. In particular, we apply SVD to the  $m$ -by- $n$  term-context matrix  $M$  produced by the SIMPLE method, resulting in the truncated matrices  $M \approx U_k \Sigma_k V_k^T$ , where  $U_k$  is a  $m$ -by- $k$  matrix,  $\Sigma_k$  is  $k$ -by- $k$ , and  $V_k$  is  $n$ -by- $k$ . This takes place after contextualization, so the component-wise operators apply in the original semantic space. Afterwards, the contextualized vector in the original space,  $\vec{c}$ , can be transformed into a vector in the reduced space,  $\vec{c}^R$ , by  $\vec{c}^R = \Sigma_k^{-1} V_k^T \vec{c}$ .

Distributional semantic vectors are traditionally compared by measures which ignore vector magnitudes, such as cosine similarity, but a multivariate Gaussian is sensitive to magnitudes. Thus, the final step is to normalize  $\vec{c}^R$  into a unit vector by dividing it by its L2 norm,  $\|\vec{c}^R\|$ .

We model the emission of these contextualized vectors in DSHMM as multivariate Gaussian distributions, so the semantic vector emissions can be written as  $P_{sem}^H, P_{sem}^A \sim \mathcal{N}(\mu, \Sigma)$ , where  $\mu \in \mathbb{R}^k$  is the mean and  $\Sigma \in \mathbb{R}^{k \times k}$  is the covariance matrix. To avoid overfitting, we regularize the covariance using its conjugate prior, the Inverse-Wishart distribution. We follow the “neutral” setting of hyperparameters given by Ornoneit and Tresp (1995), so that the MAP estimate for the covariance matrix for (event or slot) state  $i$  becomes:

$$\Sigma_i = \frac{\sum_j r_{ij}(x_j - \mu_i)(x_j - \mu_i)^T + \beta I}{\sum_j r_{ij} + 1}, \quad (8)$$

where  $j$  indexes all the relevant semantic vectors  $x_j$  in the training set,  $r_{ij}$  is the posterior responsibility of state  $i$  for vector  $x_j$ , and  $\beta$  is the remaining hyperparameter that we tune to adjust the amount of regularization. To further reduce model complexity, we set the off-diagonal entries of the resulting covariance matrix to zero.

### 3.2 Training and Inference

Inference in DSHMM is accomplished by the standard Inside-Outside and tree-Viterbi algorithms, except that the tree structure is fixed, so there is no need to sum over all possible subtrees. Model parameters are learned by the Expectation-Maximization (EM) algorithm. We tune the hyperparameters  $(N_E, N_S, \delta, \beta, k)$  and the number of EM iterations by two-fold cross-validation<sup>1</sup>.

### 3.3 Summary and Generative Process

In summary, the following steps are applied to train a DSHMM:

1. Train a distributional semantic model on a large, domain-general corpus.
2. Preprocess and generate contextualized vectors of event heads and arguments in the small corpus in the target domain.
3. Train the DSHMM using the EM algorithm.

The formal generative process is as follows:

1. Draw categorical distributions  $P_{init}^E$ ;  $P^E, P^S, P_{lemm}^H$  (one per event state);  $P_{lemm}^A$  (one per slot state) from Dirichlet priors.
2. Draw multivariate Gaussians  $P_{sem}^H, P_{sem}^A$  for each event and slot state, respectively.

<sup>1</sup>The topic cluster splits and the hyperparameter settings are available at <http://www.cs.toronto.edu/~jcheung/dshmm/dshmm.html>.

3. Generate the documents, clause by clause.

Generating a clause at position  $t$  consists of these steps:

1. Generate the event state  $E_t \sim P^E$  (or  $P_{init}^E$ ).
2. Generate the event head components  $lemm(H_t) \sim P_{lemm}^H, sem(H_t) \sim P_{sem}^H$ .
3. Generate a number of slot states  $S_{ta} \sim P^S$ .
4. For each slot, generate the argument components  $lemm(A_{ta}) \sim P_{lemm}^A, sem(A_{ta}) \sim P_{sem}^A$ .

## 4 Experiments

We trained the distributional semantic models using the Annotated Gigaword corpus (Napoles et al., 2012), which has been automatically preprocessed and is based on Gigaword 5th edition. This corpus contains almost ten million news articles and more than 4 billion tokens. We used those articles marked as “stories” — the vast majority of them. We modelled the 50,000 most common lemmata as target words, and the 3,000 most common lemmata as context words.

We then trained DSHMM and conducted our evaluations on the TAC 2010 guided summarization data set (Owczarzak and Dang, 2010). Lemmatization and extraction of event heads and arguments are done by preprocessing with the Stanford CoreNLP tool suite (Toutanova et al., 2003; de Marneffe et al., 2006). This data set contains 46 topic clusters of 20 articles each, grouped into five topic categories or domains. For example, one topic cluster in the ATTACK category is about the *Columbine Massacre*. Each topic cluster contains eight human-written “model” summaries (“model” here meaning a gold standard). Half of the articles and model summaries in a topic cluster are used in the guided summarization task, and the rest are used in the update summarization task.

We chose this data set because it allows us to conduct various domain-modelling evaluations. First, templates for the domains are provided, and the model summaries are annotated with slots from the template, allowing for an intrinsic evaluation of slot induction (Section 5). Second, it contains multiple domain instances for each of the domains, and each domain instance comes annotated with eight model summaries, allowing for an extrinsic evaluation of our system (Section 6).

## 5 Guided Summarization Slot Induction

We first evaluated our models on their ability to produce coherent clusters of entities belonging to the same slot, adopting the experimental procedure of Cheung et al. (2013).

As part of the official TAC evaluation procedure, model summaries were manually segmented into *contributors*, and labelled with the slot in the TAC template that the contributor expresses. For example, a summary fragment such as *On 20 April 1999, a massacre occurred at Columbine High School* is segmented into the contributors: (*On 20 April 1999*, WHEN); (*a massacre occurred*, WHAT); and (*at Columbine High School*, WHERE).

In the slot induction evaluation, this annotation is used as follows. First, the maximal noun phrases are extracted from the contributors and clustered based on the TAC slot of the contributor. These clusters of noun phrases then become the gold standard clusters against which automatic systems are compared. Noun phrases are considered to be matched if the lemmata of their head words are the same and they are extracted from the same summary. This accounts for the fact that human annotators often only label the first occurrence of a word that belongs to a slot in a summary, and follows the standard evaluation procedure in previous information extraction tasks, such as MUC-4. Pronouns and demonstratives are ignored. This extraction process is noisy, because the meaning of some contributors depends on an entire verb phrase, but we keep this representation to allow a direct comparison to previous work.

Because we are evaluating unsupervised systems, the clusters produced by the systems are not labelled, and must be matched to the gold standard clusters. This matching is performed by mapping to each gold cluster the best system cluster according to F1. The same system cluster may be mapped multiple times, because several TAC slots can overlap. For example, in the NATURAL DISASTERS domain, an *earthquake* may fit both the WHAT slot as well as the CAUSE slot, because it generated a *tsunami*.

We trained a DSHMM separately for each of the five domains with different semantic models, tuning hyperparameters by two-fold cross-validation. We then extracted noun phrase clusters from the model summaries according to the slot labels produced by running the Viterbi algorithm on them.

Method	P	R	F1
HMM w/o semantics	13.8	64.1	22.6*
DSHMM w/ SIMPLE	20.9	27.5	23.7
DSHMM w/ E&P	20.7	27.9	23.8
PROFINDER	<b>23.7</b>	25.0	24.3
DSHMM w/ M&L+	19.7	<b>36.3</b>	25.6*
DSHMM w/ M&L×	22.1	33.2	<b>26.5*</b>

Table 2: Slot induction results on the TAC guided summarization data set. Asterisks (\*) indicate that the model is statistically significantly different from PROFINDER in terms of F1 at  $p < 0.05$ .

**Results** We compared DSHMM to two baselines. Our first baseline is PROFINDER, a state-of-the-art template inducer which Cheung et al. (2013) showed to outperform the previous heuristic clustering method of Chambers and Jurafsky (2011). Our second baseline is our DSHMM model, without the semantic vector component, (HMM w/o semantics). To calculate statistical significance, we use the paired bootstrap method, which can accommodate complex evaluation metrics like F1 (Berg-Kirkpatrick et al., 2012).

Table 2 shows that performance of the models. Overall, PROFINDER significantly outperforms the HMM baseline, but not any of the DSHMM models by F1. DSHMM with contextualized semantic vectors achieves the highest F1s, and are significantly better than PROFINDER. All of the differences in precision and recall between PROFINDER and the other models are significant. The baseline HMM model has highly imbalanced precision and recall. We think this is because the model is unable to successfully produce coherent clusters, so the best-case mapping procedure during evaluation picked large clusters that have high recall. PROFINDER has slightly higher precision, which may be due to its non-parametric split-merge heuristic. We plan to investigate whether this learning method could improve DSHMM’s performance further. Importantly, the contextualization of the vectors seems to be beneficial, at least with the M&L component-wise operators. In the next section, we show that the improvement from contextualization transfers to multi-document summarization results.

## 6 Multi-document Summarization: An Extrinsic Evaluation

We next evaluated our models extrinsically in the setting of extractive, multi-document summarization. To use the trained DSHMM for extractive summarization, we need a decoding procedure for selecting sentences in the source text to include in the summary. Inspired by the KLSUM and HIRSUM methods of Haghighi and Vanderwende (2009), we develop a criterion based on Kullback-Leibler (KL) divergence between distributions estimated from the source text, and those estimated from the summary. The assumption here is that these distributions should match in a good summary. We describe two methods to use this criterion: a basic unsupervised method (Section 6.1), and a supervised variant that makes use of in-domain summaries to learn the salient slots and events in the domain (Section 6.2).

### 6.1 A KL-based Criterion

There are four main component distributions from our model that should be considered during extraction: (1) the distribution of events, (2) the distribution of slots, (3) the distribution of event heads, and (4) the distribution of arguments. We estimate (1) as the context-independent probability of being in a certain event state, which can be calculated using the Inside-Outside algorithm. Given a collection of documents  $D$  which make up the source text, the distribution of event topics  $\hat{P}^E(E)$  is estimated as:

$$\hat{P}^E(E = e) = \frac{1}{Z} \sum_{d \in D} \sum_t \frac{In_t(e)Out_t(e)}{P(d)}, \quad (9)$$

where  $In_t(e)$  and  $Out_t(e)$  are the values of the inside and outside trellises at timestep  $t$  for some event state  $e$ , and  $Z$  is a normalization constant. The distribution for a set of sentences in a candidate summary,  $\hat{Q}^E(E)$ , is identical, except the summation is over the clauses in the candidate summary. Slot distributions  $\hat{P}^S(S)$  and  $\hat{Q}^S(S)$  (2) are defined analogously, where the summation occurs along all the slot variables.

For (3) and (4), we simply use the MLE estimates of the lemma emissions, where the estimates are made over the source text and the candidate summary instead of over the entire training set. All of the candidate summary distributions (i.e., the “ $\hat{Q}$  distributions”) are smoothed by

a small amount, so that the KL-divergence is always finite. Our KL criterion combines the above components linearly, weighting the lemma distributions by the probability of their respective event or slot state:

$$\begin{aligned} KLScore = & \quad (10) \\ & \mathcal{D}_{KL}(\hat{P}^E || \hat{Q}^E) + \mathcal{D}_{KL}(\hat{P}^S || \hat{Q}^S) \\ & + \sum_{e=1}^{N_E} \hat{P}^E(e) \mathcal{D}_{KL}(\hat{P}^H(H|e) || \hat{Q}^H(H|e)) \\ & + \sum_{s=1}^{N_S} \hat{P}^S(s) \mathcal{D}_{KL}(\hat{P}^A(A|s) || \hat{Q}^A(A|s)) \end{aligned}$$

To produce a summary, sentences from the source text are greedily added such that  $KLScore$  is minimized at each step, until the desired summary length is reached, discarding sentences with fewer than five words.

### 6.2 Supervised Learning

The above unsupervised method results in summaries that closely mirror the source text in terms of the event and slot distributions, but this ignores the fact that not all such topics should be included in a summary. It also ignores genre-specific, stylistic considerations about characteristics of good summary sentences. For example, Woodsend and Lapata (2012) find several factors that indicate sentences should *not* be included in an extractive summary, such as the presence of personal pronouns. Thus, we implemented a second method, in which we modify the KL criterion above by estimating  $\hat{P}^E$  and  $\hat{P}^S$  from other model summaries that are drawn from the same domain (i.e. topic category), except for those summaries that are written for the specific topic cluster to be used for evaluation.

### 6.3 Method and Results

We used the best performing models from the slot induction task and the above unsupervised and supervised methods based on KL-divergence to produce 100-word summaries of the guided summarization source text clusters. We did not compare against PROFINDER, as its structure is different and would have required a different procedure than the KL-criterion we developed above. As shown in the previous evaluation, however, the HMM baseline without semantics and DSHMM with SIMPLE perform similarly in terms of F1,

Method	ROUGE-1		ROUGE-2		ROUGE-SU4	
	<i>unsup.</i>	<i>sup.</i>	<i>unsup.</i>	<i>sup.</i>	<i>unsup.</i>	<i>sup.</i>
Leading baseline	28.0	–	5.39	–	8.6	–
HMM w/o semantics	32.3	32.7	<b>6.45</b>	6.49	10.1	10.2
DSHMM w/ SIMPLE	32.1	32.7	5.81	6.50	9.8	10.2
DSHMM w/ M&L+	32.1	33.4	6.27	6.82	10.0	10.6
DSHMM w/ M&L×	32.4	<b>34.3*</b>	6.35	7.11 <sup>^</sup>	10.2	<b>11.0*</b>
DSHMM w/ E&P	<b>32.8</b>	33.8*	6.38	<b>7.31*</b>	<b>10.3</b>	10.8*

Table 3: TAC 2010 summarization results by three settings of ROUGE. Asterisks (\*) indicate that the model is statistically significantly better than the HMM model without semantics at a 95% confidence interval, a caret ^ indicates that the value is marginally so.

so we consider these competitive baselines. We did not evaluate with the update summarization task, because our method has not been adapted to it. For the evaluation measure, we used the standard ROUGE suite of automatic evaluation measures (Lin, 2004). Note that the evaluation conditions of TAC 2010 are different, and thus those results are not directly comparable to ours. For instance, top performing systems in TAC 2010 make use of manually constructed lists of entities known to fit the slots in the provided templates and sample topic statements, which our method automatically learns. We include the leading baseline results from the competition as a point of reference, as it is a well-known and non-trivial one for news articles. This baseline summary consists of the leading sentences from the most recent document in the source text cluster up to the word length limit.

Table 3 shows the summarization results for the three most widely-used settings of ROUGE. All of our models outperform the leading baseline by a large margin, demonstrating the effective of the KL-criterion. In terms of unsupervised performance, all of our models perform similarly. Because the unsupervised method mimics the distributions in the source text at all levels, the method may negate the benefit of learning and simply produce summaries that match the source text in the word distributions, thus being an approximation of KLSUM. Looking at the supervised results, however, the semantic vector models show clear gains in ROUGE, whereas the baseline method does not obtain much benefit from supervision. As in the previous evaluation, the models with contextualized semantic vectors provide the best performance. M&L× performs very well, as in slot induction, but E&P also performs well, unlike in

the previous evaluation. This result reinforces the importance of the contextualization procedure for distributional semantic models.

**Analysis** To better understand what is gained by supervision using in-domain summaries, we analyzed the best performing M&L× model’s output summaries for one document cluster from each domain. For each event state, we calculated the ratio  $\hat{P}_{summ}^E(e)/\hat{P}_{source}^E(e)$ , for the probability of an event state  $e$  as estimated from the training summaries and the the source text respectively. Likewise, we calculated  $\hat{P}_{summ}^S(s)/\hat{P}_{source}^S(s)$  for the slot states. This ratio indicates the change in state’s probability after supervision; the greater the ratio, the more preferred that state becomes after training. We selected the most preferred and dis-preferred event and slot for each document cluster, and took the three most probable lemmata from the associated lemma distribution (Table 4). It seems that supervision is beneficial because it picks out important event heads and arguments in the domain, such as *charge*, *trial*, and *murder* in the TRIALS domain. It also helps the summarizer avoid semantically generic words (*be* or *have*), pronouns, quotatives, and common but irrelevant words (*home*, *city*, *restaurant* in TRIALS).

## 7 Conclusion

We have shown that contextualized distributional semantic vectors can be successfully integrated into a generative probabilistic model for domain modelling, as demonstrated by improvements in slot induction and multi-document summarization. The effectiveness of our model stems from the use of a large domain-general corpus to train the distributional semantic vectors, and the implicit syntactic and word sense information pro-

Domain	Event Heads		Slot Arguments	
	+	-	+	-
ATTACKS	<i>say</i> <sup>2</sup> , <i>cause</i> , <i>doctor</i>	<i>say</i> <sup>2</sup> , <i>be</i> , <i>have</i>	<i>attack</i> , <i>hostage</i> , <i>troops</i>	<i>he</i> , <i>it</i> , <i>they</i>
TRIALS	<i>charge</i> , <i>trial</i> , <i>accuse</i>	<i>say</i> , <i>be</i> , <i>have</i>	<i>prison</i> , <i>murder</i> , <i>charge</i>	<i>home</i> , <i>city</i> , <i>restau- rant</i>
RESOURCES	<i>reduce</i> , <i>increase</i> , <i>university</i>	<i>say</i> , <i>be</i> , <i>have</i>	<i>government</i> , <i>effort</i> , <i>program</i>	<i>he</i> , <i>they</i> , <i>it</i>
DISASTERS	<i>flood</i> , <i>strengthen</i> , <i>engulf</i>	<i>say</i> , <i>be</i> , <i>have</i>	<i>production</i> , <i>statoil</i> , <i>barrel</i>	<i>he</i> , <i>it</i> , <i>they</i>
HEALTH	<i>be</i> , <i>department</i> , <i>have</i>	<i>say</i> , <i>do</i> , <i>make</i>	<i>food</i> , <i>product</i> , <i>meat</i>	<i>she</i> , <i>people</i> , <i>way</i>

Table 4: Analysis of the most probable event heads and arguments in the most preferred (+) and dispreferred (-) events and slots after supervised training.

vided by the contextualization process. Our approach is modular, and allows principled training of the probabilistic model using standard techniques. While we have focused on the overall clustering of entities and the distribution of event and slot topics in this work, we would also like to investigate discourse modelling and content structuring. Finally, our work shows that the application of distributional semantics to NLP tasks need not be confined to lexical disambiguation. We would like to see modern distributional semantic methods incorporated into an even greater variety of applications.

## Acknowledgments

This work is supported by the Natural Sciences and Engineering Research Council of Canada.

## References

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*.

Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. 2012. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, July. Association for Computational Linguistics.

<sup>2</sup>The event head *say* happens to appear in both the most preferred and dispreferred events in the ATTACKS domain.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3.

Asli Celikyilmaz and Dilek Hakkani-Tur. 2010. A hybrid hierarchical model for multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 815–824, Uppsala, Sweden, July. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA, June. Association for Computational Linguistics.

Jackie Chi Kit Cheung and Gerald Penn. 2012. Evaluating distributional models of semantics for syntactically invariant inference. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–43, Avignon, France, April. Association for Computational Linguistics.

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. 2008. A compositional distributional model of meaning. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*.

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

Georgiana Dinu and Mirella Lapata. 2010. Measuring distributional similarity in context. In *Proceed-*

- ings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 1162–1172.
- Micha Elsner, Joseph Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, Rochester, New York, April. Association for Computational Linguistics.
- Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 897–906.
- Elena Filatova, Vasileios Hatzivassiloglou, and Kathleen McKeown. 2006. Automatic creation of domain templates. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 207–214, Sydney, Australia, July. Association for Computational Linguistics.
- Pascale Fung and Grace Ngai. 2006. One story, one flow: Hidden markov story models for multilingual multidocument summarization. *ACM Transactions on Speech and Language Processing (TSLP)*, 3(2):1–16.
- Pascale Fung, Grace Ngai, and Chi-Shun Cheung. 2003. Combining optimal clustering and hidden markov models for extractive summarization. In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*, pages 21–28, Sapporo, Japan, July. Association for Computational Linguistics.
- Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Amit Gruber, Michael Rosen-Zvi, and Yair Weiss. 2007. Hidden topic markov models. *Artificial Intelligence and Statistics (AISTATS)*.
- Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, Boulder, Colorado, June. Association for Computational Linguistics.
- Chin Y. Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In Stan Szpakowicz and Marie-Francine Moens, editors, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.
- Annie Louis and Ani Nenkova. 2012. A coherence model based on syntactic patterns. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, July. Association for Computational Linguistics.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244.
1992. *Proceedings of the Fourth Message Understanding Conference (MUC-4)*. Morgan Kaufmann.
- Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the NAACL-HLT Joint Workshop on Automatic Knowledge Base Construction & Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100.
- Dirk Ormoneit and Volker Tresp. 1995. Improved gaussian mixture density estimates using bayesian penalty terms and network averaging. In *Advances in Neural Information Processing*, pages 542–548.
- Karolina Owczarzak and Hoa T. Dang. 2010. TAC 2010 guided summarization task guidelines.
- Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.
- Roger C. Schank and Robert P. Abelson. 1977. *Scripts, Plans, Goals, and Understanding: An Inquiry Into Human Knowledge Structures*. Lawrence Erlbaum, July.
- Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proceedings of IJCNLP*.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, page 180.
- Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Hanna M. Wallach. 2008. Structured topic models for language. *Doctoral dissertation, University of Cambridge*.
- Kristian Woodsend and Mirella Lapata. 2012. Multiple aspect summarization using integer linear programming. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, July. Association for Computational Linguistics.

# Extracting bilingual terminologies from comparable corpora

Ahmet Aker, Monica Paramita, Robert Gaizauskas  
University of Sheffield  
ahmet.aker, m.paramita, r.gaizauskas@sheffield.ac.uk

## Abstract

In this paper we present a method for extracting bilingual terminologies from comparable corpora. In our approach we treat bilingual term extraction as a classification problem. For classification we use an SVM binary classifier and training data taken from the EUROVOC thesaurus. We test our approach on a held-out test set from EUROVOC and perform precision, recall and f-measure evaluations for 20 European language pairs. The performance of our classifier reaches the 100% precision level for many language pairs. We also perform manual evaluation on bilingual terms extracted from English-German term-tagged comparable corpora. The results of this manual evaluation showed 60-83% of the term pairs generated are exact translations and over 90% exact or partial translations.

## 1 Introduction

Bilingual terminologies are important for various applications of human language technologies, including cross-language information search and retrieval, statistical machine translation (SMT) in narrow domains and computer-aided assistance to human translators. Automatic construction of bilingual terminology mappings has been investigated in many earlier studies and various methods have been applied to this task. These methods may be distinguished by whether they work on parallel or comparable corpora, by whether they assume monolingual term recognition in source and target languages (what Moore (2003) calls symmetrical approaches) or only in the source (asymmetric approaches), and by the extent to which they rely on linguistic knowledge as opposed to simply statistical techniques.

We focus on techniques for bilingual term extraction from comparable corpora – collections of source-target language document pairs that are not direct translations but are topically related. We

choose to focus on comparable corpora because for many less widely spoken languages and for technical domains where new terminology is constantly being introduced, parallel corpora are simply not available. Techniques that can exploit such corpora to deliver bilingual terminologies are of significant practical interest in these cases.

The rest of the paper is structured as follows. In Section 2 we outline our method. In Section 3 we review related work on bilingual term extraction. Section 4 describes feature extraction for term pair classification. In Section 5 we present the data used in our evaluations and discuss our results. Section 6 concludes the paper.

## 2 Method

The method we present below for bilingual term extraction is a symmetric approach, i.e. it assumes a method exists for monolingual term extraction in both source and target languages. We do not prescribe what a term must be. In particular we do not place any particular syntactic restrictions on what constitutes an allowable term, beyond the requirement that terms must be contiguous sequences of words in both source and target languages.

Our method works by first pairing each term extracted from a source language document  $S$  with each term extracted from a target language document  $T$  aligned with  $S$  in the comparable corpus. We then treat term alignment as a binary classification task, i.e. we extract features for each source-target language potential term pair and decide whether to classify the pair as a term equivalent or not. For classification purposes we use an SVM binary classifier. The training data for the classifier is derived from EUROVOC (Steinberger et al., 2002), a term thesaurus covering the activities of the EU and the European Parliament. We have run our approach on the 21 official EU languages covered by EUROVOC, constructing 20 language pairs with English as the source

language. Considering all these languages allows us to directly compare our method's performance on resource-rich (e.g. German, French, Spanish) and under-resourced languages (e.g. Latvian, Bulgarian, Estonian). We perform two different tests. First, we evaluate the performance of the classifier on a held-out term-pair list from EUROVOC using the standard measures of recall, precision and F-measure. We run this evaluation on all 20 language pairs. Secondly, we test the system's performance on obtaining bilingual terms from comparable corpora. This second test simulates the situation of using the term alignment system in a real world scenario. For this evaluation we collected English-German comparable corpora from Wikipedia, performed monolingual term tagging and ran our tool over the term tagged corpora to extract bilingual terms.

### 3 Related Work

Previous studies have investigated the extraction of bilingual terms from parallel and comparable corpora. For instance, Kupiec (1993) uses statistical techniques and extracts bilingual noun phrases from parallel corpora tagged with terms. Daille et al. (1994), Fan et al. (2009) and Okita et al. (2010) also apply statistical methods to extract terms/phrases from parallel corpora. In addition to statistical methods Daille et al. use word translation information between two words within the extracted terms as a further indicator of the correct alignment. More recently, Bouamor et al. (2012) use vector space models to align terms. The entries in the vectors are co-occurrence statistics between the terms computed over the entire corpus.

Bilingual term alignment methods that work on comparable corpora use essentially three sorts of information: (1) cognate information, typically estimated using some sort of transliteration similarity measure (2) context congruence, a measure of the extent to which the words that the source term co-occurs with have the same sort of distribution and co-occur with words with the same sort distribution as do those words that co-occur with the candidate term and (3) translation of component words in the term and/or in context words, where some limited dictionary exists. For example, in Rapp (1995), Fung and McKeown (1997), Morin et al. (2007), Cao and Li (2002) and Ismail and Manandhar (2010) the context of text units is used to identify term mappings. Transliteration and cognate-based information is exploited in Al-

Onaizan and Knight (2002), Knight and Graehl (1998), Udupa et al. (2008) and Aswani and Gaizauskas (2010).

Very few approaches have treated term alignment as a classification problem suitable for machine learning (ML) techniques. So far as we are aware, only Cao and Li (2002), who treat only base noun phrase (NP) mapping, consider the problem this way. However, it naturally lends itself to being viewed as a classification task, assuming a symmetric approach, since the different information sources mentioned above can be treated as features and each source-target language potential term pairing can be treated as an instance to be fed to a binary classifier which decides whether to align them or not. Our work differs from that of Cao and Li (2002) in several ways. First they consider only terms consisting of noun-noun pairs. Secondly for a given source language term  $\langle N_1, N_2 \rangle$ , target language candidate terms are proposed by composing all translations (given by a bilingual dictionary) of  $N_1$  into the target language with all translations of  $N_2$ . We remove both these restrictions. By considering all terms proposed by monolingual term extractors we consider terms that are syntactically much richer than noun-noun pairs. In addition, the term pairs we align are not constrained by an assumption that their component words must be translations of each other as found in a particular dictionary resource.

### 4 Feature extraction

To align or map source and target terms we use an SVM binary classifier (Joachims, 2002) with a linear kernel and the trade-off between training error and margin parameter  $c = 10$ . Within the classifier we use language dependent and independent features described in the following sections.

#### 4.1 Dictionary based features

The dictionary based features are language dependent and are computed using bilingual dictionaries which are created with GIZA++ (Och and Ney, 2000; Och and Ney, 2003). The DGT-TM parallel data (Steinberger et al., 2012) was input to GIZA++ to obtain the dictionaries. Dictionary entries have the form  $\langle s, t_i, p_i \rangle$ , where  $s$  is a source word,  $t_i$  is the  $i$ -th translation of  $s$  in the dictionary and  $p_i$  is the probability that  $s$  is translated by  $t_i$ , the  $p_i$ 's summing to 1 for each  $s$  in the dictionary. From the dictionaries we removed all entries with  $p_i < 0.05$ . In addition we also removed

every entry from the dictionary where the source word was less than four characters and the target word more than five characters in length and vice versa. This step is performed to try to eliminate translation pairs where a stop word is translated into a non-stop word. After performing these filtering steps we use the dictionaries to extract the following language dependent features:

- ***isFirstWordTranslated*** is a binary feature indicating whether the first word in the source term is a translation of the first word in the target term. To address the issue of compounding, e.g. for languages like German where what is a multi-word term in English may be expressed as a single compound word, we check whether the compound source term has an initial prefix that matches the translation of the first target word, provided that translation is at least 5 character in length.
- ***isLastWordTranslated*** is a binary feature indicating whether the last word in the source term is a translation of the last word in the target term. As with the previous feature in case of compound terms we check whether the source term ends with the translation of the target last word.
- ***percentageOfTranslatedWords*** returns the percentage of words in the source term which have their translations in the target term. To address compound terms we check for each source word translation whether it appears anywhere within the target term.
- ***percentageOfNotTranslatedWords*** returns the percentage of words of the source term which have no translations in the target term.
- ***longestTranslatedUnitInPercentage*** returns the ratio of the number of words within the longest contiguous sequence of source words which has a translation in the target term to the length of the source term, expressed as a percentage. For compound terms we proceed as with ***percentageOfTranslatedWords***.
- ***longestNotTranslatedUnitInPercentage*** returns the percentage of the number of words within the longest sequence of source words which have no translations in the target term.

These six features are direction-dependent and are computed in both directions, reversing which language is taken as the source and which as

the target. We also compute another feature *averagePercentageOfTranslatedWords* which builds the average between the feature values of *percentageOfTranslatedWords* from source to target and target to source. Thus in total we have 13 dictionary based features. Note for non-compound terms if we compare two words for equality we do not perform string match but rather use the Levenshtein Distance (see Section 4.2) between the two words and treat them as equal if the Levenshtein Distance returns  $\geq 0.95$ . This is performed to capture words with morphological differences. We set 0.95 experimentally.

## 4.2 Cognate based features

Dictionaries mostly fail to return translation entries for named entities (NEs) or specialized terminology. Because of this we also use cognate based methods to perform the mapping between source and target words or vice versa. Aker et al. (2012) have applied (1) Longest Common Subsequence Ratio, (2) Longest Common Substring Ratio, (3) Dice Similarity, (4) Needleman-Wunsch Distance and (5) Levenshtein Distance in order to extract parallel phrases from comparable corpora. We adopt these measures within our classifier. Each of them returns a score between 0 and 1.

- **Longest Common Subsequence Ratio (LCSR):** The *longest common subsequence* (LCS) measure measures the longest common non-consecutive sequence of characters between two strings. For instance, the words “dollars” and “dolari” share a sequence of 5 non-consecutive characters in the same ordering. We make use of dynamic programming (Cormen et al., 2001) to implement LCS, so that its computation is efficient and can be applied to a large number of possible term pairs quickly. We normalize relative to the length of the longest term:

$$LCSR(X, Y) = \frac{\text{len}[LCS(X, Y)]}{\max[\text{len}(X), \text{len}(Y)]}$$

where *LCS* is the longest common subsequence between two strings and characters in this subsequence need not be contiguous. The shorthand *len* stands for *length*.

- **Longest Common Substring Ratio (LCSTR):** The *longest common substring* (LCST) measure is similar to the LCS measure, but measures the longest common

*consecutive* string of characters that two strings have in common. I.e. given two terms we need to find the longest character  $n$ -gram the terms share. The formula we use for the LCSTR measure is a ratio as in the previous measure:

$$LCSTR(X, Y) = \frac{\text{len}[LCST(X, Y)]}{\max[\text{len}(X), \text{len}(Y)]}$$

- **Dice Similarity:**

$$dice = \frac{2 * LCST}{\text{len}(X) + \text{len}(Y)}$$

- **Needleman Wunsch Distance (NWD):**

$$NWD = \frac{LCST}{\min[\text{len}(X) + \text{len}(Y)]}$$

- **Levenshtein Distance (LD):** This method computes the minimum number of operations necessary to transform one string into another. The allowable operations are insertion, deletion, and substitution. Compared to the previous methods, which all return scores between 0 and 1, this method returns a score  $s$  that lies between 0 and  $n$ . The number  $n$  represents the maximum number of operations to convert an arbitrarily dissimilar string to a given string. To have a uniform score across all cognate methods we normalize  $s$  so that it lies between 0 and 1, subtracting from 1 to convert it from a distance measure to a similarity measure:

$$LD_{normalized} = 1 - \frac{LD}{\max[\text{len}(X), \text{len}(Y)]}$$

### 4.3 Cognate based features with term matching

The cognate methods assume that the source and target language strings being compared are drawn from the same character set and fail to capture the corresponding terms if this is not the case. For instance, the cognate methods are not directly applicable to the English-Bulgarian and English-Greek language pairs, as both the Bulgarian and Greek alphabets, which are Cyrillic-based, differ from the English Latin-based alphabet. However, the use of distinct alphabets is not the only problem when comparing source and target terms. Although most EU languages use the Latin alphabet, the occurrence of special characters and diacritics, as well spelling and phonetic variations,

are further challenges which are faced by term or entity mapping methods, especially in determining the variants of the same mention of the entity (Snae, 2007; Karimi et al., 2011).<sup>1</sup> We address this problem by mapping a source term to the target language writing system or vice versa. For mapping we use simple character mappings between the writing systems, such as  $\alpha \rightarrow a$ ,  $\phi \rightarrow ph$ , etc., from Greek to English. The rules allow one character on the lefthand side (source language) to map onto one or more characters on the righthand side (target language). We created our rules manually based on sound similarity between source and target language characters. We created mapping rules for 20 EU language pairs using primarily Wikipedia as a resource for describing phonetic mappings to English.

After mapping a term from source to target language we apply the cognate metrics described in 4.2 to the resulting mapped term and the original term in the other language. Since we perform both target to source and source to target mapping, the number of cognate feature scores on the mapped terms is 10 – 5 due to source to target mapping and 5 due to target to source mapping.

### 4.4 Combined features

We also combined dictionary and cognate based features. The combined features are as follows:

- ***isFirstWordCovered*** is a binary feature indicating whether the first word in the source term has a translation (i.e. has a translation entry in the dictionary regardless of the score) or transliteration (i.e. if one of the cognate metric scores is above 0.7<sup>2</sup>) in the target term. The threshold 0.7 for transliteration similarity is set experimentally using the training data. To do this we iteratively ran feature extraction, trained the classifier and recorded precision on the training data using a threshold value chosen from the interval  $[0, 1]$  in steps of 0.1. We selected as final threshold value, the lowest value for which the precision score was the same as when the threshold value was set to 1.
- ***isLastWordCovered*** is similar to the previous feature one but indicates whether the last word in the source term has a translation or

<sup>1</sup>Assuming the terms are correctly spelled, otherwise the misspelling is another problem.

<sup>2</sup>Note that we use the cognate scores obtained on the character mapped terms.

transliteration in the target term. If this is the case, 1 is returned otherwise 0.

- ***percentageOfCoverage*** returns the percentage of source term words which have a translation or transliteration in the target term.
- ***percentageOfNonCoverage*** returns the percentage of source term words which have neither a translation nor transliteration in the target term.
- ***difBetweenCoverageAndNonCoverage*** returns the difference between the last two features.

Like the dictionary based features, these five features are direction-dependent and are computed in both directions – source to target and target to source, resulting in 10 combined features.

In total we have 38 features – 13 features based on dictionary translation as described in Section 4.1, 5 cognate related features as outlined in Section 4.2, 10 cognate related features derived from character mappings over terms as described in Section 4.3 and 10 combined features.

## 5 Experiments

### 5.1 Data Sources

In our experiments we use two different data resources: EUROVOC terms and comparable corpora collected from Wikipedia.

#### 5.1.1 EUROVOC terms

EUROVOC is a term thesaurus covering the activities of the EU and the European Parliament in particular. It contains 6797 term entries in 24 different languages including 22 EU languages and Croatian and Serbian (Steinberger et al., 2002).

#### 5.1.2 Comparable Corpora

We also built comparable corpora in the information technology (IT) and automotive domains by gathering documents from Wikipedia for the English-German language pair. First, we manually chose one seed document in English as a starting point for crawling in each domain<sup>3</sup>. We then identified all articles to which the seed document is linked and added them to the crawling queue. This process is performed recursively for each document in the queue. Since our aim is to build a comparable corpus, we only added English

<sup>3</sup>[http://en.wikipedia.org/wiki/Information\\_technology](http://en.wikipedia.org/wiki/Information_technology) for IT and [http://en.wikipedia.org/wiki/Automotive\\_industry](http://en.wikipedia.org/wiki/Automotive_industry) for automotive domain.

documents which have an inter-language link in Wikipedia to a German document. We set a maximum depth of 3 in the recursion to limit size of the crawling set, i.e. documents are crawled only if they are within 3 clicks of the seed documents. A score is then calculated to represent the importance of each document  $d_i$  in this domain:

$$score_{d_i} = \sum_{j=1}^n \frac{freq_{d_{ij}}}{depth_{d_j}}$$

where  $n$  is the total number of documents in the queue,  $freq_{d_{ij}}$  is 1 if  $d_i$  is linked to  $d_j$ , or 0 otherwise, and  $depth_{d_j}$  is the number of clicks between  $d_j$  and the seed document. After all documents in the queue were assigned a score, we gathered the top 1000 documents and used inter-language link information to extract the corresponding article in the target language.

We pre-processed each Wikipedia article by performing monolingual term tagging using TWSC (Pinnis et al., 2012). TWSC is a term extraction tool which identifies terms ranging from one to four tokens in length. First, it POS-tags each document. For German POS-tagging we use TreeTagger (Schmid, 1995). Next, it uses term grammar rules, in the form of sequences of POS tags or non-stop words, to identify candidate terms. Finally, it filters the candidate terms using various statistical measures, such as pointwise mutual information and TF\*IDF.

### 5.2 Performance test of the classifier

To test the classifier’s performance we evaluated it against a list of positive and negative examples of bilingual term pairs using the measures of precision, recall and  $F$ -measure. We used 21 EU official languages, including English, and paired each non-English language with English, leading to 20 language pairs.<sup>4</sup> In the evaluation we used 600 positive term pairs taken randomly from the EUROVOC term list. We also created around 1.3M negative term pairs by pairing a source term with 200 randomly chosen distinct target terms. We select such a large number to simulate the real application scenario where the classifier will be confronted with a huge number of negative cases

<sup>4</sup>Note that we do not use the Maltese-English language pair, as for this pair we found that 5861 out of 6797 term pairs were identical, i.e. the English and the Maltese terms were the same. Excluding Maltese, the average number of identical terms between a non-English language and English in the EUROVOC data is 37.7 (out of a possible 6797).

Table 1: Wikipedia term pairs processed and judged as positive by the classifier.

	Processed	Positive
DE IT	11597K	3249
DE Automotive	12307K	1772

and a relatively small number of positive pairs. The 600 positive examples contain 200 single term pairs (i.e. single word on both sides), 200 term pairs with a single word on only one side (either source or target) and 200 term pairs with more than one word on each side. For training we took the remaining 6200 positive term pairs from EU-ROVOC and constructed another 6200 term pairs as negative examples, leading to total of 12400 term pairs. To construct the 6200 negative examples we used the 6200 terms on the source side and paired each source term with an incorrect target term. Note that we ensure that in both training and testing the set of negative and positive examples do not overlap. Furthermore, we performed data selection for each language pair separately. This means that the same pairs found in, e.g., English-German are not necessarily the same as in English-Italian. The reason for this is that the translation lengths, in number of words, vary between language pairs. For instance *adult education* is translated into *Erwachsenenbildung* in German and contains just a single word (although compound). The same term is translated into *istruzione degli adulti* in Italian and contains three words. For this reason we carry out the data preparation process separately for each language pair in order to obtain the three term pair sets consisting of term pairs with only a single word on each side, term pairs with a single word on just one side and term pairs with multiple words on both sides.

### 5.3 Manual evaluation

For this evaluation we used the Wikipedia comparable corpora collected for the English-German (EN-DE) language pair. For each pair of Wikipedia articles we used the terms tagged by TWSC and aligned each source term with every target term. This means if both source and target articles contain 100 terms then this leads to 10K term pairs. We extracted features for each pair of terms and ran the classifier to decide whether the pair is positive or negative. Table 1 shows the number of term pairs processed and the count of pairs classified as positive. Table 2 shows five

positive term pairs extracted from the English-German comparable corpora for each of the IT and automotive domains. We manually assessed a subset of the positive examples. We asked human assessors to categorize each term pair into one of the following categories:

1. **Equivalence:** The terms are exact translations/transliterations of each other.
2. **Inclusion:** Not an exact translation/transliteration, but an exact translation/transliteration of one term is entirely contained within the term in the other language, e.g: “F1 car racing” vs “Autorennen (car racing)”.
3. **Overlap:** Not category 1 or 2, but the terms share at least one translated/transliterated word, e.g: “hybrid electric vehicles” vs “hybride bauteile (hybrid components)”.
4. **Unrelated:** No word in either term is a translation/transliteration of a word in the other.

In the evaluation we randomly selected 300 pairs for each domain and showed them to two German native speakers who were fluent in English. We asked the assessors to place each of the term pair into one of the categories 1 to 4.

## 5.4 Results and Discussion

### 5.4.1 Performance test of the classifier

The results of the classifier evaluation are shown in Table 3. The results show that the overall performance of the classifier is very good. In many cases the precision scores reach 100%. The lowest precision score is obtained for Lithuanian (LT) with 67%. For this language we performed an error analysis. In total there are 221 negative examples classified as positive. All these terms are multi-term, i.e. each term pair contains at least two words on each side. For the majority of the misclassified terms – 209 in total – 50% or more of the words on one side are either translations or cognates of words on the other side. Of these, 187 contained 50% or more translation due to cognate words – examples of such cases are *capital increase* – *kapitalo eksportas* or *Arab organisation* – *Arabu lyga* with the cognates *capital* – *kapitalo* and *Arab* – *Arabu* respectively. For the remainder, 50% or more of the words on one side are dictionary translations of words on the other side. In order to understand the reason why the classifier treats such cases as positive we examined the

Table 2: Example positive pairs for English-German.

IT	Automotive
chromatographic technique — chromatographie methode	distribution infrastructure — versorgungsinfrastruktur
electrolytic capacitor — elektrolytkondensatoren	ambient temperature — außenlufttemperatur
natural user interfaces — natürliche benutzerschnittstellen	higher cetane number — erhöhter cetanzahl
anode voltage — anodenspannung	fuel tank — kraftstoffpumpe
digital subscriber loop — digitaler teilnehmeranschluss	hydrogen powered vehicle — wasserstoff fahrzeug

Table 3: Classifier performance results on EUROVOC data (P stands for precision, R for recall and  $F$  for  $F$ -measure). Each language is paired with English. The test set contains 600 positive and 1359400 negative examples.

	ET	HU	NL	DA	SV	DE	LV	FI	PT	SL	FR	IT	LT	SK	CS	RO	PL	ES	EL	BG
P	1	1	.98	1	1	.98	1	1	.7	1	1	1	.67	.81	1	1	1	1	1	1
R	.67	.72	.82	.69	.81	.77	.78	.65	.82	.66	.66	.7	.77	.84	.72	.78	.69	.8	.78	.79
$F$	.80	.83	.89	.81	.89	.86	.87	.78	.75	.79	.79	.82	.71	.91	.83	.87	.81	.88	.87	.88

training data and found 467 positive pairs which had the same characteristics as the negative examples in the testing set classified. We removed these 467 entries from the training set and re-trained the classifier. The results with the new classifier are 99% precision, 68% recall and 80%  $F$  score.

In addition to Lithuanian, two further languages, Portuguese (PT) and Slovak (SK), also had substantially lower precision scores. For these languages we also removed positive entries falling into the same problem categories as the LT ones and trained new classifiers with the filtered training data. The precision results increased substantially for both PT and SK – 95% precision, 76% recall, 84%  $F$  score for PT and 94% precision, 72% recall, 81%  $F$  score for SK. The recall scores are lower than the precision scores, ranging from 65% to 84%. We have investigated the recall problem for FI, which has the lowest recall score at 65%. We observed that all the missing term pairs were not cognates. Thus, the only way these terms could be recognized as positive is if they are found in the GIZA++ dictionaries. However, due to data sparsity in these dictionaries this did not happen in these cases. For these term pairs either the source or target terms were not found in the dictionaries. For instance, for the term pair *offshoring* — *uudelleensijoittautuminen* the GIZA++ dictionary contains the entry *offshoring* but according to the dictionary it is not translated into *uudelleensijoittautuminen*, which is the matching term in EUROVOC.

#### 5.4.2 Manual evaluation

The results of the manual evaluation are shown in Table 4. From the results we can see that both assessors judge above 80% of the IT domain terms as category 1 – the category containing equivalent

Table 4: Results of the EN-DE manual evaluation by two annotators. Numbers reported per category are percentages.

Domain	Ann.	1	2	3	4
IT	P1	81	6	6	7
	P2	83	7	7	3
Automotive	P1	66	12	16	6
	P2	60	15	16	9

term pairs. Only a small proportion of the term pairs are judged as belonging to category 4 (3–7%) – the category containing unrelated term pairs. For the automotive domain the proportion of equivalent term pairs varies between 60 and 66%. For unrelated term pairs this is below 10% for both assessors.

We investigated the inter-annotator agreement. Across the four classes the percentage agreement was 83% for the automotive domain term pairs and 86% for the IT domain term pairs. The kappa statistic,  $\kappa$ , was .69 for the automotive domain pairs and .52 for the IT domain. We also considered two class agreement where we treated term pairs within categories 2 and 3 as belonging to category 4 (i.e. as “incorrect” translations). In this case, for the automotive domain the percentage agreement was 90% and  $\kappa = 0.72$  and for the IT domain percentage agreement was 89% with  $\kappa = 0.55$ . The agreement in the automotive domain is higher than in the IT one although both judges were computer scientists. We analyzed the differences and found that they differ in cases where the German and the English term are both in English. One of the annotators treated such cases as correct translation, whereas the other did not.

We also checked to ensure our technique was not simply rediscovering our dictionaries. Since the GIZA++ dictionaries contain only single word–single word mappings, we examined the

newly aligned term pairs that consisted of one word on both source and target sides. Taking both the IT and automotive domains together, our algorithm proposed 5021 term pairs of which 2751 (55%) were word-word term pairs. 462 of these (i.e. 17% of the word-word term pairs or 9% of the overall set of aligned term pairs) were already in either the EN-DE or DE-EN GIZA++ dictionaries. Thus, of our newly extracted term pairs a relatively small proportion are rediscovered dictionary entries. We also checked our evaluation data to see what proportion of the assessed term pairs were already to be found in the GIZA++ dictionaries. A total of 600 term pairs were put in front of the judges of which 198 (33%) were word-word term pairs. Of these 15 (less than 8% of the word-word pairs and less than 3% of the overall assessed set of assessed term pairs) were word-word pairs already in the dictionaries. We conclude that our evaluation results are not unduly affected by assessing term pairs which were given to the algorithm.

**Error analysis** For both domains we performed an error analysis for the unrelated, i.e. category 4 term pairs. We found that in both domains the main source of errors is due to terms with different meanings but similar spellings such as the following example (1).

- (1) *accelerator* — *decelerator*

For this example the cognate methods, e.g. the Levenshtein similarity measure, returns a score of 0.81. This problem could be addressed in different ways. First, it could be resolved by applying a very high threshold for the cognate methods. Any cognate score below that threshold could be regarded as zero – as we did for the combined features (cf. Section 4.4). However, setting a similarity threshold higher than 0.9 – to filter out cases as in (1) – will cause real cognates with greater variation in the spellings to be missed. This will, in particular, affect languages with a lot of inflection, such as Latvian. Another approach to address this problem would be to take the contextual or distributional properties of the terms into consideration. To achieve this, training data consisting of term pairs along with contextual information is required. However, such training data does not currently exist (i.e. resources like EUROVOC do not contain contextual information) and it would need to be collected as a first step towards applying this approach to the problem.

**Partial Translation** The assessors assigned 6 – 7% of the term pairs in the IT domain and 12 – 16% in the automotive domain to categories 2 and 3. In both categories the term pairs share translations or cognates.

Clearly, if humans such as professional translators are the end users of these terms, then it could be helpful for them to find some translation units within the terms. In category 2 this will be the entire translation of one term in the other such as the following examples.<sup>5</sup>

- (2) *visible graphical interface* — *grafische benutzerschnittstelle*  
 (3) *modern turbocharger systems* — *moderne turbolader*

In example (3) the a translation of the German term is to be found entirely within in the English term but the English term has the additional word *visible*, a translation of which is not found in the German term. In example (4), again the translation of the German term is entirely found in the English term, but as in the previous example, one of the English words – *systems* – in this case, has no match within the German term. In category 3 there are only single word translation overlaps between the terms as shown in the following examples.

- (4) *national standard language* — *niederländischen standardsprache*  
 (5) *thermoplastic material* — *thermoplastische elastomere*

In example (5) *standard language* is translated to *standardsprache* and in example (6) *thermoplastic* to *thermoplastische*. The other words within the terms are not translations of each other.

Another application of the extracted term pairs is to use them to enhance existing parallel corpora to train SMT systems. In this case, including the partially correct terms may introduce noise. This is especially the case for the terms within category 3. However, the usefulness of terms in both these scenarios requires further investigation, which we aim to do in future work.

<sup>5</sup>In our data it is always the case that the target term is entirely translated within the English one and the other way round.

## 6 Conclusion

In this paper we presented an approach to align terms identified by a monolingual term extractor in bilingual comparable corpora using a binary classifier. We trained the classifier using data from the EUROVOC thesaurus. Each candidate term pair was pre-processed to extract various features which are cognate-based or dictionary-based. We measured the performance of our classifier using Information Retrieval (IR) metrics and a manual evaluation. In the IR evaluation we tested the performance of the classifier on a held out test set taken from EUROVOC. We used 20 EU language pairs with English being always the source language. The performance of our classifier in this evaluation reached the 100% precision level for many language pairs. In the manual evaluation we had our algorithm extract pairs of terms from Wikipedia articles – articles forming comparable corpora in the IT and automotive domains – and asked native speakers to categorize a selection of the term pairs into categories reflecting the level of translation of the terms. In the manual evaluation we used the English-German language pair and showed that over 80% of the extracted term pairs were exact translations in the IT domain and over 60% in the automotive domain. For both domains over 90% of the extracted term pairs were either exact or partial translations.

We also performed an error analysis and highlighted problem cases, which we plan to address in future work. Exploring ways to add contextual or distributional features to our term representations is also an avenue for future work, though it clearly significantly complicates the approach, one of whose advantages is its simplicity. Furthermore, we aim to extend the existing dictionaries and possibly our training data with terms extracted from comparable corpora. Finally, we plan to investigate the usefulness of the terms in different application scenarios, including computer assisted translation and machine translation.

## Acknowledgements

The research reported was funded by the TaaS project, European Union Seventh Framework Programme, grant agreement no. 296312. The authors would like to thank the manual annotators for their helpful contributions. We would also like to thank partners at Tilde SIA and at the University of Zagreb for supplying the TWSC term extraction

tool, developed within the EU funded project AC-CURAT.

## References

- A. Aker, Y. Feng, and R. Gaizauskas. 2012. Automatic bilingual phrase extraction from comparable corpora. In *24th International Conference on Computational Linguistics (COLING 2012)*, IIT Bombay, Mumbai, India, 2012. Association for Computational Linguistics.
- Y. Al-Onaizan and K. Knight. 2002. Machine transliteration of names in arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, pages 1–13. Association for Computational Linguistics.
- N. Aswani and R. Gaizauskas. 2010. English-hindi transliteration using multiple similarity metrics. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, Valetta, Malta.
- D. Bouamor, N. Semmar, and P. Zweigenbaum. 2012. Identifying bilingual multi-word expressions for statistical machine translation. In *LREC 2012, Eighth International Conference on Language Resources and Evaluation*, pages 674–679, Istanbul, Turkey, 2012. ELRA.
- Y. Cao and H. Li. 2002. Base noun phrase translation using web data and the em algorithm. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. 2001. *Introduction to Algorithms*. The MIT Press, 2nd revised edition, September.
- B. Daille, É. Gaussier, and J.M. Langé. 1994. Towards automatic extraction of monolingual and bilingual terminology. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 515–521. Association for Computational Linguistics.
- X. Fan, N. Shimizu, and H. Nakagawa. 2009. Automatic extraction of bilingual terms from a chinese-japanese parallel corpus. In *Proceedings of the 3rd International Universal Communication Symposium*, pages 41–45. ACM.
- P. Fung and K. McKeown. 1997. Finding terminology translations from non-parallel corpora. In *Proceedings of the 5th Annual Workshop on Very Large Corpora*, pages 192–202.
- A. Ismail and S. Manandhar. 2010. Bilingual lexicon extraction from comparable corpora using in-domain terms. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 481–489. Association for Computational Linguistics.

- T. Joachims. 2002. *Learning to classify text using support vector machines: Methods, theory and algorithms*, volume 186. Kluwer Academic Publishers Norwell, MA, USA:.
- S. Karimi, F. Scholer, and A. Turpin. 2011. Machine transliteration survey. *ACM Computing Surveys (CSUR)*, 43(3):17.
- K. Knight and J. Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4):599–612.
- J. Kupiec. 1993. An algorithm for finding noun phrase correspondences in bilingual corpora. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 17–22. Association for Computational Linguistics.
- R. Moore. 2003. Learning translations of named-entity phrases from parallel corpora. In *In Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 259–266. Association for Computational Linguistics.
- E. Morin, B. Daille, K. Takeuchi, and K. Kageura. 2007. Bilingual terminology mining - using brain, not brawn comparable corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 664–671, Prague, Czech Republic, June. Association for Computational Linguistics.
- F. J. Och and H. Ney. 2000. A comparison of alignment models for statistical machine translation. In *Proceedings of the 18th conference on Computational linguistics*, pages 1086–1090, Morristown, NJ, USA. Association for Computational Linguistics.
- F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- T. Okita, A. Maldonado Guerra, Y. Graham, and A. Way. 2010. Multi-word expression-sensitive word alignment. Association for Computational Linguistics.
- Mărcis Pinnis, Nikola Ljubešić, Dan Ștefănescu, Ingun Skadiņa, Marko Tadić, and Tatiana Gornostay. 2012. Term extraction, tagging, and mapping tools for under-resourced languages. In *Proc. of the 10th Conference on Terminology and Knowledge Engineering (TKE 2012)*, June, pages 20–21.
- R. Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 320–322. Association for Computational Linguistics.
- Helmut Schmid. 1995. Treetagger— a language independent part-of-speech tagger. *Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart*, page 43.
- C. Snae. 2007. A comparison and analysis of name matching algorithms. *International Journal of Applied Science, Engineering and Technology*, 4(1):252–257.
- R. Steinberger, B. Pouliquen, and J. Hagman. 2002. Cross-lingual document similarity calculation using the multilingual thesaurus eurovoc. *Computational Linguistics and Intelligent Text Processing*, pages 101–121.
- R. Steinberger, A. Eisele, S. Klocek, S. Pilos, and P. Schlter. 2012. Dgt-tm: A freely available translation memory in 22 languages. In *Proceedings of LREC*, pages 454–459.
- R. Udupa, K. Saravanan, A. Kumaran, and J. Jagarlamudi. 2008. Mining named entity transliteration equivalents from comparable corpora. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1423–1424. ACM.

# *The Haves and the Have-Nots: Leveraging Unlabelled Corpora for Sentiment Analysis*

Kashyap Popat<sup>2</sup> Balamurali A R<sup>1,2,3</sup> Pushpak Bhattacharyya<sup>2</sup> Gholamreza Haffari<sup>3</sup>

<sup>1</sup>IITB-Monash Research Academy, IIT Bombay

<sup>3</sup>Monash University

<sup>2</sup>Dept. of Computer Science and Engineering, IIT Bombay

Australia

{kashyap, balamurali, pb}@cse.iitb.ac.in

reza@monash.edu

## Abstract

Expensive feature engineering based on WordNet senses has been shown to be useful for document level sentiment classification. A plausible reason for such a performance improvement is the reduction in data sparsity. However, such a reduction could be achieved with a lesser effort through the means of syntagma based word clustering. In this paper, the problem of data sparsity in sentiment analysis, both monolingual and cross-lingual, is addressed through the means of clustering. Experiments show that cluster based data sparsity reduction leads to performance better than sense based classification for sentiment analysis at document level. Similar idea is applied to Cross Lingual Sentiment Analysis (CLSA), and it is shown that reduction in data sparsity (after translation or bilingual-mapping) produces accuracy higher than Machine Translation based CLSA and sense based CLSA.

## 1 Introduction

Data sparsity is the bane of Natural Language Processing (NLP) (Xue et al., 2005; Minkov et al., 2007). Language units encountered in the test data but absent in the training data severely degrade the performance of an NLP task. NLP applications innovatively handle data sparsity through various means. A special, but very common kind of data sparsity *viz.*, word sparsity, can be addressed in one of the two obvious ways: 1) sparsity reduction through *paradigmatically related* words or 2) sparsity reduction through *syntagmatically related* words.

Paradigmatic analysis of text is the analysis of concepts embedded in the text (Cruse, 1986; Chandler, 2012). WordNet is a byproduct of such an analysis. In WordNet, paradigms are manually generated based on the principles of lexical and semantic relationship among words (Fellbaum, 1998). WordNets are primarily used to address the problem of word sense disambiguation. However, at present there are many NLP applications which use WordNet. One such application is Sentiment Analysis (SA) (Pang and Lee, 2002). Recent research has shown that word sense based semantic features can improve the performance of SA systems (Rentoumi et al., 2009; Tamara et al., 2010; Balamurali et al., 2011) compared to word based features.

Syntagmatic analysis of text concentrates on the surface properties of the text. Compared to paradigmatic property extraction, syntagmatic processing is relatively light weight. One of the obvious syntagmas is *words*, and words are grouped into equivalence classes or clusters, thus reducing the model parameters of a statistical NLP system (Brown et al., 1992). When used as an additional feature with word based language models, it has been shown to improve the system performance *viz.*, machine translation (Uszkoreit and Brants, 2008; Stymne, 2012), speech recognition (Martin et al., 1995; Samuelsson and Reichl, 1999), dependency parsing (Koo et al., 2008; Haffari et al., 2011; Zhang and Nivre, 2011; Tratz and Hovy, 2011) and NER (Miller et al., 2004; Faruqui and Padó, 2010; Turian et al., 2010; Täckström et al., 2012).

In this paper, the focus is on alleviating the data sparsity faced by supervised approaches for SA through the means of cluster based features. As WordNets are essentially word

clusters wherein words with the same meaning are clubbed together, they address the problem of data sparsity at word level. The abstraction and dimensionality reduction thus achieved attributes to the superior performance for SA systems that employs WordNet senses as features. However, WordNets are manually created. Automatic creation of the same is challenging and not much successful because of the linguistic complexity involved. In case of SA, manually creating the features based on WordNet senses is a tedious and an expensive process. Moreover, WordNets are not present for many languages. All these factors make the paradigmatic property based cluster features like WordNet senses a less promising pursuit for SA.

The syntagmatic analysis essentially makes use of distributional similarity and may in many circumstances subsume the paradigmatic analysis. In the current work, this particular insight is used to solve the data sparsity problem in the sentiment analysis by leveraging unlabelled monolingual corpora. Specifically, experiments are performed *to investigate whether features developed from manually crafted clusterings (coming from WordNet) can be replaced by those generated from clustering based on syntagmatic properties.*

Further, *cluster based features are used to address the problem of scarcity of sentiment annotated data in a language.* Popular approaches for Cross-Lingual Sentiment Analysis (CLSA) (Wan, 2009; Duh et al., 2011) depend on Machine Translation (MT) for converting the labeled data from one language to the other (Hiroshi et al., 2004; Banea et al., 2008; Wan, 2009). However, many languages which are truly resource scarce, do not have an MT system or existing MT systems are not ripe to be used for CLSA (Balamurali et al., 2013). To perform CLSA, this study leverages unlabelled parallel corpus to generate the word alignments. These word alignments are then used to link cluster based features to obliterate the language gap for performing SA. No MT systems or bilingual dictionaries are used for this study. Instead, language gap for performing CLSA is bridged using linked cluster or *cross-lingual* clusters (explained in section 4) with the help of unlabelled monolingual corpora. The contributions of this paper are two fold:

1. *Features created from manually built and finer clusters can be replaced by inexpensive cluster based features generated solely from unlabelled corpora.* Experiments performed on four publicly available datasets in three languages *viz.*, *English, Hindi* and *Marathi*<sup>1</sup> suggest that cluster based features can considerably boost the performance of an SA system. Moreover, state of the art result is obtained for one of the publicly available dataset.
2. *An alternative and effective approach for CLSA is demonstrated using clusters as features.* Word clustering is a powerful mechanism to “transfer” a sentiment classifier from one language to another. Thus can be used in truly resource scarce scenarios like that of *English-Marathi* CLSA.

The rest of the paper is organized as follows: section 2 presents related work. Section 3 explains different word cluster based features employed to reduce data sparsity for monolingual SA. In section 4, alternative CLSA approaches based on word clustering are elucidated. Experimental details are explained in section 5. Results and discussions are presented in section 6 and section 7 respectively. Finally, section 8 concludes the paper pointing to some future research possibilities.

## 2 Related Work

The problem of SA at document level is defined as the classification of document into different polarity classes (positive and negative) (Turney, 2002). Both supervised (Benamara et al., 2007; Martineau and Finin, 2009) and unsupervised approaches (Mei et al., 2007; Lin and He, 2009) exist for this task.

Supervised approaches are popular because of their superior classification accuracy (Mullen and Collier, 2004; Pang and Lee, 2008). Feature engineering plays an important role in these systems. Apart from the commonly used bag-of-words features based on unigrams/bigrams/ngrams (Dave et al., 2003; Ng et al., 2006; Martineau and Finin, 2009),

---

<sup>1</sup>Hindi and Marathi belong to the Indo-Aryan subgroup of the Indo-European language family and are two widely spoken Indian languages with a speaker population of 450 million and 72 million respectively.

syntax (Matsumoto et al., 2005; Nakagawa et al., 2010), semantic (Balamurali et al., 2011) and negation (Ikeda et al., 2008) have also been explored for this task. There has been research related to clustering and sentiment analysis. In Rooney et al. (2011), documents are clustered based on the context of each document and sentiment labels are attached at the cluster level. Zhai et al. (2011) attempts to cluster features of a product to perform sentiment analysis on product reviews. In this work, word clusters (syntagmatic and paradigmatic) encoding a mixture of syntactic and semantic information are used for feature engineering.

In situations where labeled data is not present in a language, approaches based on cross-lingual sentiment analysis are used. Most often these methods depend on an intermediary machine translation system (Wan, 2009; Brooke et al., 2009) or a bilingual dictionary (Ghorbel and Jacot, 2011; Lu et al., 2011) to bridge the language gap. Given the subtle and different ways the sentiment can be expressed which itself manifested as a result of cultural diversity amongst different languages, an MT system has to be of a superior quality to capture them.

### 3 Clustering for Sentiment Analysis

The goal of this paper, to remind the reader, is to investigate whether superior word cluster features based on manually crafted and fine grained lexical resource like WordNet can be replaced with the syntagmatic property based word clusters created from unlabelled monolingual corpora.

In this section, different clustering approaches are presented for feature engineering in a monolingual setting.

#### 3.1 Approach 1: Clustering based on WordNet Sense

A synonymous set of words in a WordNet is called a synset. Each synset can be considered as a word cluster comprising of semantically similar words. Balamurali et al. (2011) showed that WordNet synsets can act as good features for document level sentiment classification.

Motivation for their study stems from the fact that different senses of a word can have different polarities. To empirically prove the superiority of sense based features, different variants of a travel review domain corpus were generated

by using automatic/manual sense disambiguation techniques. Thereafter, accuracies of classifiers based on different sense-based and word-based features were compared. The results suggested that WordNet synset based features performed better than word-based features.

In this study, synset identifiers are extracted from manually/automatically sense annotated corpora and used as features for creating sentiment classifiers. The classifier thus build is used as a baseline. Apart from this, another baseline employing word based features are used for a comprehensive comparison.

#### 3.2 Approach 2: Syntagmatic Property based Clustering

For this particular study, a co-occurrence based algorithm is used to create word clusters. As the algorithm is based on co-occurrence, one can extract the classes that have the flavour of syntagmatic grouping, depending on the nature of underlying statistics. Agglomerative clustering algorithm by Brown et al. (1992) is used for this purpose. It is a hard clustering algorithm *i.e.*, each word belongs to one cluster only.

Formally, as mentioned in Brown et al. (1992), let  $C$  be a hard clustering function which maps vocabulary  $V$  to one of the  $K$  clusters. Then, the likelihood ( $L()$ ) of a sequence of word tokens,  $w = [w_j]_{j=1}^m$ , with  $w_j \in V$ , can be factored as,

$$L(w; C) = \prod_{j=1}^m p(w_j | C(w_j)) p(C(w_j) | C(w_{j-1})) \quad (1)$$

Words are assigned to clusters such that the above quantity is maximized. For the purpose of sentiment classification, cluster identifiers representing words in the document are used as features for training.

### 4 Clustering for Cross Lingual Sentiment Analysis

Existing approaches for CLSA depend on an intermediary machine translation system to bridge the language gap (Hiroshi et al., 2004; Banea et al., 2008). Machine translation is very resource intensive. If a language is truly resource scarce, it is mostly unlikely to have an MT system. Given that sentiment analysis is a less resource intensive task compared to machine translation, the use of an MT system is hard to justify for performing

CLSA. As a viable alternative, cluster linkages could be learned from a bilingual parallel corpus and these *linkages* can be used to bridge the language gap for CLSA.

In this section, three approaches using clusters as features for CLSA are compared. The language whose annotated data is used for training is called the source language ( $S$ ), while the language whose documents are to be sentiment classified is referred to as the target language ( $T$ ).

#### 4.1 Approach 1: Projection based on Sense (PS)

In this approach, a Multidict is used to bridge the language gap for SA. A Multidict is an instance of WordNet where the same sense from different languages are linked (Mohanty et al., 2008). An entry in the multidict will have a WordNet sense identifier from  $S$  and the corresponding WordNet sense identifier from  $T$ . The approach of projection based on sense is explained in Algorithm 1. Note that after the *Sense Mark* operation, each document will be represented as a vector of WordNet sense identifiers.

---

##### Algorithm 1 Projection based on sense

---

**Input:** Polarity labeled data in source language ( $S$ ) and data in target language ( $T$ ) to be labeled

**Output:** Classified documents

- 1: Sense mark the polarity labeled data from  $S$
  - 2: Project the sense marked corpora from  $S$  to  $T$  using a Multidict
  - 3: Model the sentiment classifier using the data obtained in step-2
  - 4: Sense mark the unlabelled data from  $T$
  - 5: Test the sentiment classifier on data obtained in step-4 using model obtained in step-3
- 

Sense identifiers are the features for the classifier. For those sense identifiers which do not have a corresponding entry in the Multidict, no projection is performed.

#### 4.2 Approach 2: Direct Cluster Linking (DCL)

Given a parallel bilingual corpus, word clusters in  $S$  can be aligned to clusters in  $T$ . Word alignments are created using parallel corpora. Given two aligned word sequences  $w^S = [w_j^S]_{j=1}^m$  and  $w^T = [w_k^T]_{k=1}^n$ , let  $\alpha^{T|S}$  be a set of scored alignments from the source language to the target

language. Here, an alignment from the  $a_k^{th}$  source word to the  $k^{th}$  target word, with score  $s_{k,a_k} > \epsilon$  is represented as  $(w_k^T, w_{a_k}^S, s_{k,a_k}) \in \alpha^{T|S}$ . To simplify,  $k \in \alpha^{T|S}$  is used to denote those target words  $w_k^T$  that are aligned to some source word  $w_{a_k}^S$ .

The source and the target side clusters are linked using the Equation (2).

$$LC(l) = \underset{t}{\operatorname{argmax}} \sum_{\substack{k \in \alpha^{T|S} \cup \alpha^{S|T} \\ s.t. C^T(w_k^T) = t \\ C^S(w_{a_k}^S) = l}} s_{k,a_k} \quad (2)$$

Here, a target side cluster  $t \in C^T$  is linked to a source side cluster  $l \in C^S$  such that the total alignment score between words in  $l$  and words in  $t$  is maximum.  $C^S$  and  $C^T$  stands for source and target side cluster list respectively.  $LC(l)$  gives the target side cluster  $t$  to which  $l$  is linked.

#### 4.3 Approach 3: Cross-Lingual Clustering (XC)

Direct cluster linking approach suffers from the size of alignment dataset in the form of parallel corpora. The size of the alignment dataset is typically smaller than the monolingual dataset. To circumvent this problem, Täckström et al. (2012) introduced cross-lingual clustering. In cross-lingual clustering, the objective function maximizes the joint likelihood of monolingual and cross-lingual factors. Given a list of words and clusters it belongs to, a clustering algorithm tries to obtain word-cluster association which maximizes the joint likelihood of words and clusters. Whereas in case of cross-lingual clustering, the same clustering can be explained in terms of maximizing the likelihood of monolingual word-cluster pairs of the source, the target and alignments between them.

Formally, as stated in Täckström et al. (2012), Using the model of Uszkoreit and Brants (2008), the likelihood of a sequence of word tokens,  $w = [w_j]_{j=1}^m$ , with  $w_j \in V$ , can be factored as,

$$L(w; C) = \prod_{j=1}^m p(w_j | C(w_j)) p(C(w_j) | w_{j-1}) \quad (3)$$

Note this is different from the likelihood estimation of Brown et al. (1992) (Equation (1)), where  $C(w_j)$  was conditioned on  $C(w_{j-1})$ . This

makes the computation easier as suggested in the original paper. The Equation (3) in a cross lingual setting will be transformed as given below:

$$L^{S,T}(w^S, w^T; \alpha^{T|S}, \alpha^{S|T}, C^S, C^T) = L^S(\dots).L^T(\dots).L^{T|S}(\dots).L^{S|T}(\dots) \quad (4)$$

Here,  $L^{T|S}(\dots)$  and  $L^{S|T}(\dots)$  are factors based on word alignments, which can be represented as:

$$L^{T|S}(w^T; \alpha^{T|S}, C^T, C^S) = \prod_{k \in \alpha^{T|S}} p(w_k^T | C^T(w_k^T)) p(C^T(w_k^T) | C^S(w_{a_k}^S)) \quad (5)$$

Based on the optimization objective in Equation (4), a pseudo algorithm is defined in Algorithm 2. For more information, readers are requested to refer Täckström et al. (2012).

---

**Algorithm 2** Cross-lingual Clustering (XC)

---

**Input:** Source and target language corpus

**Output:** Cross-lingual clusters

- 1: ##  $C^S, C^T$  randomly initialized
  - 2: **for**  $i \leftarrow 1$  to  $N$  **do**
  - 3: Find  $C_*^S \approx \operatorname{argmax}_{C^S} L^S(w^S; C^S)$
  - 4: Project  $C_*^S$  to  $C^T$
  - 5: Find  $C_*^T \approx \operatorname{argmax}_{C^T} L^T(w^T; C^T)$
  - 6: Project  $C_*^T$  to  $C^S$
  - 7: **end for**
- 

An MT based CLSA approach is used as the baseline. Training data from  $S$  is translated to  $T$  and classification model is learned using unigram based features. Thereafter, the classifier is directly tested on data from  $T$ .

## 5 Experimental Setup

Analysis was performed on three languages, *viz.*, *English (En)*, *Hindi (Hi)* and *Marathi (Mar)*. CLSA was performed on two language pairs, *English-Hindi* and *English-Marathi*. For clustering the words, monolingual data of Indian Languages Corpora Initiative (ILCI)<sup>2</sup> was used. It should also be noted that sentiment annotated data was also included in the data used for the word clusterings process. For Brown clustering, an implementation by Liang (2005) was used. Cross-lingual clustering for CLSA

<sup>2</sup><http://sanskrit.jnu.ac.in/ilci/index.jsp>

was implemented as directed in Täckström et al. (2012).

**Monolingual SA:** For experiments in *English*, two polarity datasets were used. The first one (*En-TD*) by Ye et al. (2009) contains user-written reviews on travel destinations. The dataset consists of approximately 600 positive and 591 negative reviews. Reviews were also manually sense annotated using WordNet 2.1. The sense annotation was performed by two annotators with an inter-annotation agreement of 93%. The second dataset (*En-PD*)<sup>3</sup> on product reviews (music instruments) from Amazon by Blitzer et al. (2007) contains 1000 positive and 1000 negative reviews. This dataset was sense annotated using an automatic WSD engine which was trained on tourism domain (Khapra et al., 2010). Experiments using this dataset were done to study the effect of domain on CLSA. For experiments in *Hindi* and *Marathi*, polarity datasets by Balamurali et al. (2012) were used.<sup>4</sup> These are reviews collected from various *Hindi* and *Marathi* blogs and Sunday editorials. *Hindi* dataset consist of 98 positive and 100 negative reviews. Whereas *Marathi* dataset contains 75 positive and 75 negative reviews. Apart from being marked with polarity labels at document level, they are also manually sense annotated using *Hindi* and *Marathi* WordNet respectively.

**CLSA:** The same datasets used in SA are also used for CLSA. Three approaches (as described in section 4) were tested for *English-Hindi* and *English-Marathi* language pairs. To create alignments, *English-Hindi* and *English-Marathi* parallel corpora from ILCI were used. *English-Hindi* parallel corpus contains 45992 sentences and *English-Marathi* parallel corpus contains 47881 sentences. To create alignments, GIZA++<sup>5</sup> was used (Och and Ney, 2003).

As a preprocessing step, all stop words were removed. Stemming was performed on English and Hindi whereas for Marathi data, Morphological Analyzer was used to reduce the words to their respective lemmas.

All experiments were performed using C-SVM

<sup>3</sup><http://www.cs.jhu.edu/~mdredze/datasets/sentiment/>

<sup>4</sup>[http://www.cfilt.iitb.ac.in/resources/senti/MPLC\\_tour\\_downloaderInfo.php](http://www.cfilt.iitb.ac.in/resources/senti/MPLC_tour_downloaderInfo.php)

<sup>5</sup><http://www-i6.informatik.rwth-aachen.de/Colleagues/och/software/GIZA++.html>

Features	En-TD	En-PD	Hi	Mar
Words	87.02	77.60	77.36	92.28
WordNet Sense (Paradigmatic)	89.13	74.50	<b>85.80</b>	96.88
Clusters (Syntagmatic)	<b>97.45</b>	<b>87.80</b>	83.50 <sup>x</sup>	<b>98.66</b>

Table 1: Classification accuracy for monolingual sentiment analysis. For English, results are reported on two publicly available datasets based on Travel Domain (TD) and Product Domain (PD).

Features	Words	Clust-200	Clust-500	Clust-1000	Clust-1500	Clust-2000	Clust-2500	Clust-3000
En-TD	87.02	97.37	97.45	96.94	96.94	96.52	96.52	96.52
En-PD	77.60	73.20	82.30	84.30	86.35	86.45	87.80	87.40

Table 2: Classification accuracy (in %) versus cluster size (number of clusters to be used).

(linear kernel with parameter optimized over training set using 5 fold cross validation) available as a part of LibSVM package<sup>6</sup>. SVM was used since it is known to perform well for sentiment classification (Pang et al., 2002). Results reported are based on the average of ten-fold cross-validation accuracies. Standard text metrics are used for reporting the experimental results.

## 6 Results

Monolingual classification results are shown in Table<sup>7</sup>1. Table shows accuracies of SA systems developed on feature set based on words, senses and clusters. It must be noted that accuracies reported for cluster based features are with respect to the best accuracy based on different cluster sizes. The improvements in results of cluster features based approach is found to be statistically significant over the word features based approach and sense features based approach at 95% confidence level when tested using a paired t-test (except for *Hindi* cluster features based approach). But in general, their accuracies do not significantly vary after cluster size crosses 1500.

Table 2 shows the classification accuracy variation when cluster size is altered. For, En-TD and En-PD experiments, the cluster size was varied between 200-3000 with an interval of 500 (after a size of 500). In the En-TD experiment, the best accuracy is achieved for cluster size 500, which is lesser than the number of unique-words/unique-senses (6435/6004) present in the data. Similarly, for the En-PD experiment,

the optimal cluster size of 2500 is also lesser than the number of unique-words/unique-senses (30468/4735) present in the data.

To see the effect of training data size variation for different SA approaches in the En-TD experiment, the training data size is varied between 50 to 500. For this, a test set consisting of 100 positive and 100 negative documents is fixed. The training data size is varied by selecting different number of documents from rest of the dataset (~500 negative and ~500 positive) as a training set. For each training data set 10 repeats are performed, *e.g.*, for training data size of 50, 50 negative and 50 positive documents are randomly selected from the training data pool of ~500 negative and ~500 positive. This was repeated 10 times (with replacement). The results of this experiment are presented in Figure 1.

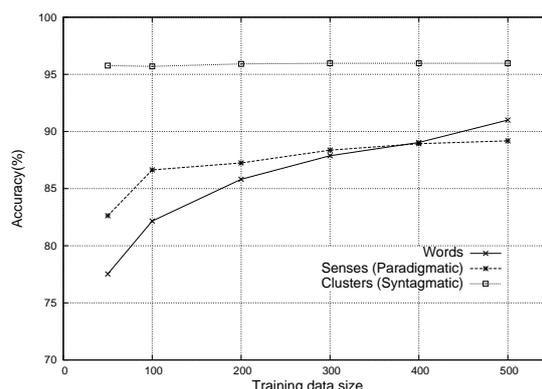


Figure 1: Training data variation on En-TD dataset.

Cross-lingual SA accuracies are presented in Table 3. As in monolingual case, the reported accuracies are for features based on the best cluster size.

<sup>6</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvm>

<sup>7</sup>All results reported here are based on 10-fold except for Marathi (2-fold-5-repeats), as it had comparatively lesser data samples.

Target Language	MT	PS	DCL	XC
<i>T=Hi</i>	63.13	53.80	51.51	<b>66.16</b>
<i>T=Mar</i>	NA	54.00	56.00	<b>60.30</b>

Table 3: Cross-Lingual SA accuracy (%) on *T=Hi* and *T=Mar* with *S=En* for different approaches (MT=Machine Translation, PS=Projection based on Sense, DCL=Direct Cluster Linking, XC=Cross-Lingual Clustering. There is no MT system available for (*S=En, T=Mar*).

## 7 Discussions

In this section, some important observations from the results are discussed.

**1. Syntagmatic analysis may be used in lieu of paradigmatic analysis for SA:** The results suggest that word cluster based features using syntagmatic analysis is comparatively better than cluster (sense) based features using paradigmatic analysis. For two datasets in *English* and for the one in *Marathi* this holds true. For *English*, the gap between classification accuracy based on sense features and cluster features is around 10%. A state-of-art accuracy is obtained for the public dataset on travel domain (*En-TD*).

The difference in accuracy reduces as the language gets morphologically rich. In a morphologically rich language, morphology encompasses syntactical information, limiting the context it can provide for clustering. This can be seen from the classification results on *Marathi*. However for *Hindi*, classifier built on features based on syntagmatic analysis trails the one based on paradigmatic analysis.

Compared to *Marathi*, *Hindi* is a less morphologically rich language, hence, a better result was expected. However, a contrary result was obtained.<sup>8</sup> In *Hindi*, the subject and the object of the sentence are linked using a case marker. Upon error analysis, it was found that there was a lot of irregular compounding based on case markers. Case markers were compounded with the succeeding word. This is a deviation from the real scenario which would have resulted in incorrect clustering leading to an unexpected result. However, the same would not have occurred for a classifier developed on sense based features as it was manually sense tagged.

Clustering induces a reduction in the data sparsity. For example, on *En-PD*, percentage of features present in the test set and not present in the training set to those present in the test set are 34.17%, 11.24%, 0.31% for words, synsets

and cluster based features respectively. The improvement in the performance of classifiers may be attributed to this feature size reduction. However, it must be noted that clustering based on unlabelled corpora is less taxing than manually creating paradigmatic property based clusters like WordNet synsets.

Barring one instance, both cluster based features outperform word based features. The reason for the drop in the accuracy of approach based on sense features for *En-PD* dataset is the domain specific nature of sentiment analysis (Blitzer et al., 2007), which is explained in the next point.

**2. Domain issues are resolved while using cluster based features:** For *En-PD*, the classifier developed using sense features based on paradigmatic analysis performs inferior to word based features. Compared to other datasets used for analysis, this dataset was sense annotated using an automatic WSD engine. This engine was trained on a travel domain corpus and as WSD is also domain specific, the final classification performance suffered. Additionally, as the target domain was on products, the automatic WSD engine employed had an in-domain accuracy of 78%. The sense disambiguation accuracy of the same would have lowered in a cross-domain setting. This might have had a degrading effect on the SA accuracy.

However, it was seen that classifier developed on cluster features based on syntagmatic analysis do not suffer from this. Such clusters obliterate domain related issues. In addition, as more unlabelled data is included for clustering, the classification accuracy improves.<sup>8</sup> Thus, clustering may be employed to tackle other specific domain related issues in SA.

<sup>8</sup>It was observed that adding 0.1 million unlabelled documents, SA accuracy improved by 1%. This was observed in the case of English for which there is abundant unlabelled corpus.

**3. Cluster based features using syntagmatic analysis requires lesser training data:** Cluster based features drastically reduces the dimension of the feature vector. For instance, the size of sense based features for En-TD dataset was  $1/6^{th}$  of the size of word based features. This reduces the perplexity of the classification model. The reduction in the perplexity leads to the reduction of training documents to attain the same classification accuracy without any dimensionality reduction. This is evident from Figure 1 where accuracy of the cluster features based on unlabelled corpora are higher even with lesser training data.

**4. Effect of cluster size:** The cluster size (number of clusters employed) has an implication on the purity of each cluster with respect to the application. The system performance improved upon increasing the cluster size and converged after attaining a certain level of accuracy. In general, it was found that the best classification accuracy was obtained for a cluster size between 1000 and 2500. As evident from Table 2, once the optimal accuracy is obtained, no significant changes were observed by increasing the cluster size.

**5. Clustering based CLSA is effective:** For target language as *Hindi*, CLSA accuracy based on cross-lingual clustering (syntagmatic) outperforms the one based on MT (refer to Table 3). This was true for the constraint clustering approach based on cross-lingual clustering. Whereas, sentiment classifier using sense (PS) or direct cluster linking (DCL) is not very effective. In case of PS approach, the coverage of the mult字典 was a problem. The number of a linkages between sense from *English* to *Hindi* is only around  $1/3^{rd}$  the size of Princeton WordNet (Fellbaum, 1998). Similarly in case of DCL approach, monolingual likelihood is different from the cross-lingual likelihood in terms of the linkages.

**6. A note on CLSA for truly resource scarce languages:** Note that there is no publicly available MT system for *English* to *Marathi*. Moreover, the digital content in *Marathi* language does not have a standard encoding format. This impedes the automatic crawling of the web for corpora creation for SA. Much manual effort has to be put to collect enough corpora for analysis. However, even in these languages, unlabelled corpora is

easy to obtain. *Marathi* was chosen to depict a truly resource scarce SA scenario. Cluster features based classifier comparatively performed well with 60% classification accuracy. An MT based system would have suffered in this case as *Marathi*, as stated earlier, is a morphologically rich language and as compared to English, has a different word ordering. This could degrade the accuracy of the machine translation itself, limiting the performance of an MT based CLSA system. All this is obliterated by the use of a cluster based CLSA approach. Moreover, as more monolingual corpora is added for clustering, the cross lingual cluster linkages could be refined. This can further boost the CLSA accuracy.

## 8 Conclusion and Future Work

This paper explored feasibility of using word cluster based features in lieu of features based on WordNet senses for sentiment analysis to alleviate the problem of data sparsity. Abstractly, the motivation was to see if highly effective features based on paradigmatic property based clustering could be replaced with the inexpensive ones based on syntagmatic property for SA.

The study was performed for both monolingual SA and cross-lingual SA. It was found that cluster features based on syntagmatic analysis are better than the WordNet sense features based on paradigmatic analysis for SA. Investigation revealed that a considerable decrease in the training data could be achieved while using such class based features. Moreover, as syntagma based word clusters are homogenous, it was able to address domain specific nature of SA as well.

For CLSA, clusters linked together using unlabelled parallel corpora do away with the need of translating labelled corpora from one language to another using an intermediary MT system or bilingual dictionary. Such a method outperforms an MT based CLSA approach. Further, this approach was found to be useful in cases where there are no MT systems to perform CLSA and the language of analysis is truly resource scarce. Thus, wider implication of this study is that many widely spoken yet resource scarce languages like *Pashto*, *Sundanese*, *Hausa*, *Gujarati* and *Punjabi* which do not have an MT system could now be analysed for sentiment. The approach presented here for CLSA will still require a parallel corpora. However, the size of the parallel corpora required

for CLSA can considerably be much lesser than the size of the parallel corpora required to train an MT system.

A naive cluster linkage algorithm based on word alignments was used to perform CLSA. As a result, there were many erroneous linkages which lowered the final SA accuracy. Better cluster-linking approaches could be explored to alleviate this problem. There are many applications which use WordNet like IR, IE *etc.* It would be interesting to see if these could be replaced by clusters based on the syntagmatic property.

## References

- A. R. Balamurali, Aditya Joshi, and Pushpak Bhattacharyya. 2011. Harnessing wordnet senses for supervised sentiment classification. In *Proceedings of EMNLP 2011*, pages 1081–1091, Stroudsburg, PA, USA.
- A. R. Balamurali, Aditya Joshi, and Pushpak Bhattacharyya. 2012. Cross-lingual sentiment analysis for Indian languages using linked wordnets. In *Proceedings of COLING 2012*, pages 73–82, Mumbai, India.
- A. R. Balamurali, Mitesh M. Khapra, and Pushpak Bhattacharyya. 2013. Lost in translation: viability of machine translation for cross language sentiment analysis. In *Proceedings of CICLing 2013*, pages 38–49, Berlin, Heidelberg.
- Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. 2008. Multilingual subjectivity analysis using machine translation. In *Proceedings of EMNLP 2008*, pages 127–135, Honolulu, Hawaii.
- Farah Benamara, Sabatier Irit, Carmine Cesarano, Napoli Federico, and Diego Reforgiato. 2007. Sentiment analysis: Adjectives and adverbs are better than adjectives alone. In *Proceedings of the International Conference on Weblogs and Social Media*.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL 2007*, pages 440–447, Prague, Czech Republic.
- Julian Brooke, Milan Tofiloski, and Maite Taboada. 2009. Cross-linguistic sentiment analysis: From english to spanish. In *Proceedings of the International Conference RANLP-2009*, pages 50–54, Borovets, Bulgaria.
- Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, pages 467–479, December.
- D. Chandler. 2012. Semiotics for beginners. <http://users.aber.ac.uk/dgc/Documents/S4B/sem01.html>. Online, accessed 20-February-2013.
- D. A. Cruse. 1986. *Lexical Semantics*. Cambridge University Press.
- Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of WWW 2003*, pages 519–528, New York, NY, USA.
- Kevin Duh, Akinori Fujino, and Masaaki Nagata. 2011. Is machine translation ripe for cross-lingual sentiment classification? In *Proceedings of ACL-HLT 2011*, pages 429–433, Stroudsburg, PA, USA.
- Manaal Faruqui and Sebastian Padó. 2010. Training and Evaluating a German Named Entity Recognizer with Semantic Generalization. In *Proceedings of KONVENS 2010*, Saarbrücken, Germany.
- Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.
- Hatem Ghorbel and David Jacot. 2011. Further experiments in sentiment analysis of french movie reviews. In *Proceedings of AWIC 2011*, pages 19–28, Fribourg, Switzerland.
- Gholamreza Haffari, Marzieh Razavi, and Anoop Sarkar. 2011. An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *Proceedings of ACL-HLT 2011*, pages 710–714, Stroudsburg, PA, USA.
- Kanayama Hiroshi, Nasukawa Tetsuya, and Watanabe Hideo. 2004. Deeper sentiment analysis using machine translation technology. In *Proceedings of COLING 2004*, Stroudsburg, PA, USA.
- Daisuke Ikeda, Hiroya Takamura, Lev arie Ratinov, and Manabu Okumura. 2008. Learning to shift the polarity of words for sentiment classification. In *Proceedings of the Third International Joint Conference on Natural Language Processing*.
- Mitesh Khapra, Sapan Shah, Piyush Kedia, and Pushpak Bhattacharyya. 2010. Domain-specific word sense disambiguation combining corpus based and wordnet based parameters. In *Proceedings of Global Wordnet Conference*.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-HLT 2008*, pages 595–603, Columbus, Ohio.
- Percy Liang. 2005. Semi-supervised learning for natural language. M. eng. thesis, Massachusetts Institute of Technology.

- Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of CIKM 2009*, pages 375–384, New York, NY, USA.
- Bin Lu, Chenhao Tan, Claire Cardie, and Benjamin K. Tsou. 2011. Joint bilingual sentiment classification with unlabeled parallel corpora. In *Proceedings of ACL-HLT 2011*, pages 320–330, Stroudsburg, PA, USA.
- Sven Martin, Jrg Liermann, and Hermann Ney. 1995. Algorithms for bigram and trigram word clustering. In *Speech Communication*, pages 1253–1256.
- Justin Martineau and Tim Finin. 2009. Delta TFIDF: An improved feature space for sentiment analysis. In *Proceedings of ICWSM*.
- Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment classification using word sub-sequences and dependency sub-trees. In *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 301–311.
- Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of WWW 2007*, pages 171–180, New York, NY, USA.
- Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT-NAACL 2004: Main Proceedings*, pages 337–342, Boston, Massachusetts, USA.
- Einat Minkov, Kristina Toutanova, and Hisami Suzuki. 2007. Generating complex morphology for machine translation. In *Proceedings of ACL 2007*, pages 128–135, Prague, Czech Republic.
- Rajat Mohanty, Pushpak Bhattacharyya, Prabhakar Pande, Shraddha Kalele, Mitesh Khapra, and Aditya Sharma. 2008. Synset based multilingual dictionary: Insights, applications and challenges. In *Proceedings of Global Wordnet Conference*.
- Tony Mullen and Nigel Collier. 2004. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of EMNLP 2004*, pages 412–418, Barcelona, Spain.
- Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *Proceedings of HLT-NAACL 2010*, pages 786–794, Stroudsburg, PA, USA.
- Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin. 2006. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *Proceedings of the COLING 2006*, pages 611–618, Stroudsburg, PA, USA.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.
- Bo Pang and Lillian Lee. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP 2002*, pages 79–86, Stroudsburg, PA, USA.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, January.
- Vassiliki Rentoumi, George Giannakopoulos, Vangelis Karkaletsis, and George A. Vouros. 2009. Sentiment analysis of figurative language using a word sense disambiguation approach. In *Proceedings of RANLP 2009*, pages 370–375, Borovets, Bulgaria, September.
- Niall Rooney, Hui Wang, Fiona Browne, Fergal Monaghan, Jann Mller, Alan Sergeant, Zhiwei Lin, Philip Taylor, and Vladimir Dobrynin. 2011. An exploration into the use of contextual document clustering for cluster sentiment analysis. In *Proceedings of RANLP 2011*, pages 140–145, Hissar, Bulgaria.
- C. Samuelsson and W. Reichl. 1999. A class-based language model for large-vocabulary speech recognition extracted from part-of-speech statistics. In *Proceedings of ICASSP 1999*, pages 537–540.
- Sara Stymne. 2012. Clustered word classes for pre-ordering in statistical machine translation. In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 28–34.
- Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual Word Clusters for Direct Transfer of Linguistic Structure. In *Proceedings of NAACL-HLT 2012*, pages 477–487, Montréal, Canada.
- Martin Tamara, Balahur Alexandra, and Montoyo Andres. 2010. Word sense disambiguation in opinion mining: Pros and cons. *Journal Research in Computing Science*, 46:119–130.
- Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of EMNLP 2011*, pages 1257–1268, Stroudsburg, PA, USA.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL 2010*, pages 384–394, Stroudsburg, PA, USA.
- Peter D. Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL 2002*, pages 417–424, Stroudsburg, PA, USA.

- Jakob Uszkoreit and Thorsten Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *Proceedings of ACL-HLT 2008*, pages 755–762, Columbus, Ohio.
- Xiaojun Wan. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of ACL 2009*, pages 235–243, Stroudsburg, PA, USA.
- Gui-Rong Xue, Chenxi Lin, Qiang Yang, WenSi Xi, Hua-Jun Zeng, Yong Yu, and Zheng Chen. 2005. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of SIGIR 2005*, pages 114–121, New York, NY, USA.
- Qiang Ye, Ziqiong Zhang, and Rob Law. 2009. Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert Systems with Applications*, 36(3, Part 2):6527–6535.
- Zhongwu Zhai, Bing Liu, Hua Xu, and Peifa Jia. 2011. Clustering product features for opinion mining. In *Proceedings of WSDM 2011*, pages 347–354, New York, NY, USA.
- Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of ACL-HLT 2011*, pages 188–193, Stroudsburg, PA, USA.

# Large-scale Semantic Parsing via Schema Matching and Lexicon Extension

Qingqing Cai

Temple University

Computer and Information Sciences  
qingqing.cai@temple.edu

Alexander Yates

Temple University

Computer and Information Sciences  
yates@temple.edu

## Abstract

Supervised training procedures for semantic parsers produce high-quality semantic parsers, but they have difficulty scaling to large databases because of the sheer number of logical constants for which they must see labeled training data. We present a technique for developing semantic parsers for large databases based on a reduction to standard supervised training algorithms, schema matching, and pattern learning. Leveraging techniques from each of these areas, we develop a semantic parser for Freebase that is capable of parsing questions with an F1 that improves by 0.42 over a purely-supervised learning algorithm.

## 1 Introduction

Semantic parsing is the task of translating natural language utterances to a formal meaning representation language (Chen et al., 2010; Liang et al., 2009; Clarke et al., 2010; Liang et al., 2011; Artzi and Zettlemoyer, 2011). There has been recent interest in producing such semantic parsers for large, heterogeneous databases like Freebase (Krishnamurthy and Mitchell, 2012; Cai and Yates, 2013) and Yago2 (Yahya et al., 2012), which has driven the development of semi-supervised and distantly-supervised training methods for semantic parsing. Previous purely-supervised approaches have been limited to smaller domains and databases, such as the GeoQuery database, in part because of the cost of labeling enough samples to cover all of the logical constants involved in a domain.

This paper investigates a reduction of the problem of building a semantic parser to three standard problems in semantics and machine learning: supervised training of a semantic parser, schema matching, and pattern learning. Figure 1 provides a visualization of our system architecture. We apply an existing supervised training algorithm for semantic parsing to a labeled data set. We

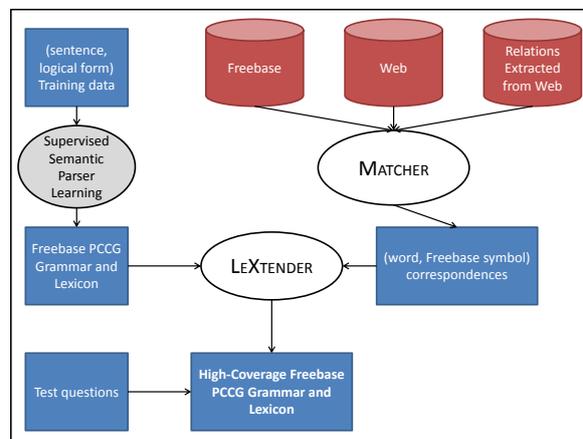


Figure 1: We reduce the task of learning a large-scale semantic parser to a combination of 1) a standard supervised algorithm for learning semantic parsers; 2) our MATCHER algorithm for finding correspondences between words and database symbols; and 3) our LEXTENDER algorithm for integrating (word, database symbol) matches into a semantic parsing lexicon.

apply schema matching techniques to the problem of finding correspondences between English words  $w$  and ontological symbols  $s$ . And we apply pattern learning techniques to incorporate new  $(w, s)$  pairs into the lexicon of the trained semantic parser.

This reduction allows us to apply standard techniques from each problem area, which in combination provide a large improvement over the purely-supervised approaches. On a dataset of 917 questions taken from 81 domains of the Freebase database, a standard learning algorithm for semantic parsing yields a parser with an F1 of 0.21, in large part because of the number of logical symbols that appear during testing but never appear during training. Our techniques can extend this parser to new logical symbols through schema matching, and yield a semantic parser with an F1 of 0.63 on the same task. On a more challenging task where training and test data are divided so that all logical constants in test are never observed dur-

ing training, our approach yields a semantic parser with an F1 of 0.6, whereas the purely supervised approach cannot parse a single test question correctly. These results indicate that it is possible to automatically extend semantic parsers to symbols for which little or no training data has been observed.

The rest of this paper is organized as follows. The next section discusses related work. Section 3 describes our MATCHER algorithm for performing schema matching between a knowledge base and text. Section 4 explains how we use MATCHER’s schema matching to extend a standard semantic parser to logical symbols for which it has seen no labeled training data. Section 5 analyzes the performance of MATCHER and our semantic parser. Section 6 concludes.

## 2 Previous Work

Two existing systems translate between natural language questions and database queries over large-scale databases. Yahya *et al.* (2012) report on a system for translating natural language queries to SPARQL queries over the Yago2 (Hofmann *et al.*, 2013) database. Yago2 consists of information extracted from Wikipedia, WordNet, and other resources using manually-defined extraction patterns. The manual extraction patterns pre-define a link between natural language terms and Yago2 relations. Our techniques automate the process of identifying matches between textual phrases and database relation symbols, in order to scale up to databases with more relations, like Freebase. A more minor difference between Yahya *et al.*’s work and ours is that their system handles SPARQL queries, which do not handle aggregation queries like `argmax` and `count`. We rely on an existing semantic parsing technology to learn the language that will translate into such aggregation queries. On the other hand, their test questions involve more conjunctions and complex semantics than ours. Developing a dataset with more complicated semantics in the queries is part of our ongoing efforts.

Krishnamurthy and Mitchell (2012) also create a semantic parser for Freebase covering 77 of Freebase’s over 2000 relations. Like our work, their technique uses distant supervision to drive training over a collection of sentences gathered from the Web, and they do not require any manually-labeled training data. However, their

technique does require manual specification of rules that construct CCG lexical entries from dependency parses. In comparison, we fully automate the process of constructing CCG lexical entries for the semantic parser by making it a prediction task. We also leverage synonym-matching techniques for comparing relations extracted from text with Freebase relations. Finally, we test our results on a dataset of 917 questions covering over 600 Freebase relations, a more extensive test than the 50 questions used by Krishnamurthy and Mitchell.

Numerous methods exist for comparing two relations based on their sets of tuples. For instance, the DIRT system (Lin and Pantel, 2001) uses the mutual information between the  $(X, Y)$  argument pairs for two binary relations to measure the similarity between them, and clusters relations accordingly. More recent examples of similar techniques include the Resolver system (Yates and Etzioni, 2009) and Poon and Domingos’s USP system (Poon and Domingos, 2009). Our techniques for comparing relations fit into this line of work, but they are novel in their application of these techniques to the task of comparing database relations and relations extracted from text.

Schema matching (Rahm and Bernstein, 2001; Ehrig *et al.*, 2004; Giunchiglia *et al.*, 2005) is a task from the database and knowledge representation community in which systems attempt to identify a “common schema” that covers the relations defined in a set of databases or ontologies, and the mapping between each individual database and the common schema. Owing to the complexity of the general case, researchers have resorted to defining standard similarity metrics between relations and attributes, as well as machine learning algorithms for learning and predicting matches between relations (Doan *et al.*, 2004; Wick *et al.*, 2008b; Wick *et al.*, 2008a; Nottelmann and Straccia, 2007; Berlin and Motro, 2006). These techniques consider only matches between relational databases, whereas we apply these ideas to matches between Freebase and extracted relations. Schema matching in the database sense often considers complex matches between relations (Dhamanka *et al.*, 2004), whereas as our techniques are currently restricted to matches involving one database relation and one relation extracted from text.

### 3 Textual Schema Matching

#### 3.1 Problem Formulation

The textual schema matching task is to identify natural language words and phrases that correspond with each relation and entity in a fixed schema for a relational database. To formalize this task, we first introduce some notation.

A *schema*  $S = (E, R, C, I)$  consists of a set of entities  $E$ , a set of relations  $R$ , a set of categories  $C$ , and a set of instances  $I$ . Categories are one-argument predicates (e.g., `film(e)`), and relations are two- (or more-) argument predicates (e.g., `directed.by(e1, e2)`). Instances are known tuples of entities that make a relation or category true, such as `film(Titanic)` or `directed.by(Titanic, James Cameron)`. For a given  $r \in R$  (or  $c \in C$ ),  $I_S(r)$  indicates the set of known instances of  $r$  in schema  $S$  (and likewise for  $I_S(c)$ ). Examples of such schemas include Freebase (Bollacker et al., 2008) and Yago2 (Hoffart et al., 2013). We say a schema is a *textual* schema if it has been extracted from free text, such as the Nell (Carlson et al., 2010) and ReVerb (Fader et al., 2011) extracted databases.

Given a textual schema  $T$  and a database schema  $D$ , the textual schema matching task is to identify an alignment or *matching*  $M \subset R_T \times R_D$  such that  $(r_T, r_D) \in M$  if and only if  $r_T$  can be used to refer to  $r_D$  in normal language usage. The problem would be greatly simplified if  $M$  were a 1-1 function, but in practice most database relations can be referred to in many ways by natural language users: for instance, `film_actor` can be referenced by the English verbs “played,” “acted,” and “starred,” along with morphological variants of them. In addition, many English verbs can refer to several different relations in Freebase: “make” can refer to `computer_processor_manufacturer` or `distilled_spirits_producer`, among many others. Our MATCHER algorithm for textual schema matching handles this by producing a confidence score for every possible  $(r_T, r_D)$  pair, which downstream applications can then use to reason about the possible alignments.

Even worse than the ambiguities in alignment, some textual relations do not correspond with any database relation exactly, but instead they correspond with a projection of a relation, or a join between multiple relations, or another com-

plex view of a database schema. As a simple example, “actress” corresponds to a subset of the Freebase `film_actor` relation that intersects with the set  $\{x: \text{gender}(x, \text{female})\}$ . MATCHER can only determine that “actress” aligns with `film_actor` or not; it cannot produce an alignment between “actress” and a join of `film_actor` and `gender`. These more complex alignments are an important consideration for future work, but as our experiments will show, quite useful alignments can be produced without handling these more complex cases.

#### 3.2 Identifying candidate matches

MATCHER uses a generate-and-test architecture for determining  $M$ . It uses a Web search engine to issue queries for a database relation  $r_D$  consisting of all the entities in a tuple  $t \in I_D(r_D)$ . 1000 tuples for each  $r_D$  are randomly chosen for issuing queries. The system then retrieves matching snippets from the search engine results. It uses the top 10 results for each search engine query. It then counts the frequency of each word type in the set of retrieved snippets for  $r_D$ . The top 500 non-stopword word types are chosen as candidates for matches with  $r_D$ . We denote the candidate set for  $r_D$  as  $C(r_D)$ .

MATCHER’s threshold of 500 candidates for  $C(r_D)$  results in a maximum possible recall of just less than 0.8 for the alignments in our dataset, but even if we double the threshold to 1000, the recall improves only slightly to 0.82. We therefore settled on 500 as a point with an acceptable upper bound on recall, while also producing an acceptable number of candidate terms for further processing.

#### 3.3 Pattern-based match selection

The candidate pool  $C(r_D)$  of 500 word types is significantly smaller than the set of all textual relations, but it is also extremely noisy. The candidates may include non-relation words, or other frequent but unrelated words. They may also include words that are highly related to  $r_D$ , but not actually corresponding textual relations. For instance, the candidate set for `film_director` in Freebase includes words like “directed,” but also words like “film,” “movie,” “written,” “produced,” and “starring.” We use a series of filters based on synonym-detection techniques to help select the true matching candidates from  $C(r_D)$ .

	Pattern	Condition	Example
1.	“ $r_T$ in $E$ ”	$r_T$ ends with “-ed” and $E$ has type <code>datetime</code> or <code>location</code>	“founded in 1989”
2.	“ $r_T$ by $E$ ”	$r_T$ ends with “-ed”	“invented by Edison”
3.	“ $r_T$ such as $E$ ”	$r_T$ ends with “-s”	“directors such as Tarantino”
4.	“ $E$ is a(n) $r_T$ ”	all cases	“Paul Rudd is an actor”

Table 1: Patterns used by MATCHER as evidence of a match between  $r_D$  and  $r_T$ .  $E$  represents an entity randomly selected from the tuples in  $I_D(r_D)$ .

The first type of evidence we consider for identifying true matches from  $C(r_D)$  consists of pattern-matching. Relation words that express  $r_D$  will often be found in complex grammatical constructions, and often they will be separated from their entity arguments by long-distance dependencies. However, over a large corpus, one would expect that in at least some cases, the relation word will appear in a simple, relatively-unambiguous grammatical construction that connects  $r_T$  with entities from  $r_D$ . For instance, entities  $e$  from the relationship `automotive_designer` appear in the pattern “designed by  $e$ ” more than 100 times as often as the next most-common patterns, “considered by  $e$ ” and “worked by  $e$ .”

MATCHER uses searches over the Web to count the number of instances where a candidate  $r_T$  appears in simple patterns that involve entities from  $r_D$ . Greater counts for these patterns yield greater evidence of a correct match between  $r_D$  and  $r_T$ . Table 1 provides a list of patterns that we consider. For each  $r_D$  and each  $r_T \in C(r_D)$ , MATCHER randomly selects 10 entities from  $r_D$ ’s tuples to include in its pattern queries. Two of the patterns are targeted at past-tense verbs, and the other two patterns at nominal relation words.

MATCHER computes statistics similar to pointwise mutual information (PMI) (Turney, 2001) to measure how related  $r_D$  and  $r_T$  are, for each pattern  $p$ . Let  $c(p, r_D, r_T)$  indicate the sum of all the counts for a particular pattern  $p$ , database relation, and textual relation:

$$f_p(r_T, r_D) = \frac{c(p, r_D, r_T)}{\sum_{r'_D} c(p, r'_D, r_T) * \sum_{r'_T} c(p, r_D, r'_T)}$$

For the sum over all  $r'_D$ , we use all  $r'_D$  in Freebase for which  $r_T$  was extracted as a candidate.

One downside of the pattern-matching evidence is the sheer number of queries it requires. Freebase

currently has over 2,000 relations. For each  $r_D$ , we have up to 500 candidate  $r_T$ , up to 4 patterns, and up to 10 entities per pattern. To cover all of Freebase, MATCHER needs  $2,000 \times 500 \times 4 \times 10 = 40$  million queries, or just over 1.25 years if it issues 1 query per second (we covered approximately one-quarter of Freebase’s relations in our experiments). Using more patterns and more entities per pattern are desirable for accumulating more evidence about candidate matches, but there is a trade-off with the time required to issue the necessary queries.

### 3.4 Comparing database relations with extracted relations

Open Information Extraction (Open IE) systems (Banko et al., 2007) can often provide a large set of extracted tuples for a given  $r_T$ , which MATCHER can then use to make much more comprehensive comparisons with the full tuple set for  $r_D$  than the pattern-matching technique allows.

MATCHER employs a form of PMI to compute the degree of relatedness between  $r_D$  and  $r_T$ . In its simplest form, MATCHER computes:

$$PMI(r_T, r_D) = \frac{|I_D(r_D) \cap I_T(r_T)|}{|I_D(r_D)| \cdot |I_T(r_T)|} \quad (1)$$

While this PMI statistic is already quite useful, we have found that in practice there are many cases where an exact match between tuples in  $I_D(r_D)$  and tuples in  $I_T(r_T)$  is too strict of a criterion. MATCHER uses a variety of approximate matches to compute variations of this statistic. Considered as predictors for the true matches in  $M$ , these variations of the PMI statistic have a lower precision, in that they are more likely to have high values for incorrect matches. However, they also have a higher recall: that is, they will have a high value for correct candidates in  $C(r_D)$  when the strict version of PMI does not. Table 2 lists all the variations used by MATCHER.

Statistics for $(r_T, r_D)$
$s_\kappa(r_T, r_D) = \frac{\sum_{t_D \in I_D(r_D)} \sum_{t_T \in I_T(r_T)} \kappa(t_D, t_T)}{ I_D(r_D)  \cdot  I_T(r_T) }$
$s'_\kappa(r_T, r_D) = \frac{s_\kappa(r_T, r_D)}{\sum_{r'_D} s_\kappa(r'_D, r_T)}$
$s''(r_T, r_D) = \frac{ I_T(r_T) }{ I_D(r_D) }$

Table 2: MATCHER statistics: for each  $\kappa$  function for comparing two tuples (given in Table 3), MATCHER computes the statistics above to compare  $r_D$  and  $r_T$ . The PMI statistic in Equation 1 corresponds to  $s_\kappa$  where  $\kappa$  =strict match over  $\Phi$  =full tuples.

$\kappa(t_1, t_2)$ for comparing tuples $t_1, t_2$
strict match: $\begin{cases} 1, \text{ if } \Phi(t_1) = \Phi'(t_2) \\ 0, \text{ otherwise.} \end{cases}$
type match: $\begin{cases} 1, \text{ if } \forall_k \text{cat}(\Phi(t_1)_k) \\ \quad = \text{cat}(\Phi'(t_2)_k) \\ 0, \text{ otherwise.} \end{cases}$

Table 3: MATCHER’s  $\kappa$  functions for computing whether two tuples are similar. *cat* maps an entity to a category (or type) in the schema. MATCHER has a different  $\kappa$  function for each possible combination of  $\Phi$  and  $\Phi'$  functions, which are given in Table 4.

MATCHER uses an API for the ReVerb Open IE system<sup>1</sup> (Fader et al., 2011) to collect  $I(r_T)$ , for each  $r_T$ . The API for ReVerb allows for relational queries in which some subset of the entity strings, entity categories, and relation string are specified. The API returns all matching triples; types must match exactly, but relation or argument strings in the query will match any relation or argument that contains the query string as a substring. MATCHER queries ReVerb with three different types of queries for each  $r_T$ , specifying the types for both arguments, or just the type of the first argument, or just the second argument. Types for arguments are taken from the types of arguments for a potentially matching  $r_D$  in Freebase. To avoid overwhelming the ReVerb servers, for our experiments we limited MATCHER to queries

<sup>1</sup><http://openie.cs.washington.edu/>

$\Phi(t)$ for tuple $t = (e_1, \dots, e_n)$
$\forall_i e_i$ (projection to one dimension)
$(e_1, \dots, e_n)$ (full tuple)
$\forall_{\sigma(\cdot)} (e_{\sigma(1)}, \dots, e_{\sigma(n)})$ (permutation)

Table 4: MATCHER’s  $\Phi$  functions for projecting or permuting a tuple.  $\sigma$  indicates a permutation of the indices.

for the top 80  $r_T \in C(r_D)$ , when they are ranked according to frequency during the candidate identification process.

### 3.5 Regression models for scoring candidates

Pattern statistics, the ReVerb statistics from Table 2, and the count of  $r_T$  during the candidate identification step all provide evidence for correct matches between  $r_D$  and  $r_T$ . MATCHER uses a regression model to combine these various statistics into a score for  $(r_T, r_D)$ . The regression model is a linear regression with least-squares parameter estimation; we experimented with support vector regression models with non-linear kernels, with no significant improvements in accuracy. Section 5 explains the dataset we use to train this model. Unlike a classifier, MATCHER does not output any single matching  $M$ . However, downstream applications can easily convert MATCHER’s output into a matching  $M$  by, for instance, selecting the top  $K$  candidate  $r_T$  values for each  $r_D$ , or by selecting all  $(r_T, r_D)$  pairs with a score over a chosen threshold. Our experiments analyze MATCHER’s success by comparing its performance across a range of different values for the number of  $r_T$  matches for each  $r_D$ .

## 4 Extending a Semantic Parser Using a Schema Alignment

An alignment between textual relations and database relations has many possible uses: for example, it might be used to allow queries over a database to be answered using additional information stored in an extracted relation store, or it might be used to deduce clusters of synonymous relation words in English. Here, we describe an application in which we build a question-answering system for Freebase by extending a standard learning technique for semantic parsing with schema alignment information.

As a starting point, we used the UBL system

developed by Kwiatkowski *et al.* (2010) to learn a semantic parser based on probabilistic Combinatory Categorical Grammar (PCCG). Source code for UBL is freely available. Its authors found that it achieves results competitive with the state-of-the-art on a variety of standard semantic parsing data sets, including Geo250 English (0.85 F1). Using a fixed CCG grammar and a procedure based on unification in second-order logic, UBL learns a lexicon  $\Lambda$  from the training data which includes entries like:

#### Example Lexical Entries

---

New York City  $\vdash NP : \text{new\_york}$   
 neighborhoods in  $\vdash$   
 $S \setminus NP / NP : \lambda x \lambda y . \text{neighborhoods}(x, y)$

#### Example CCG Grammar Rules

---

$X/Y : f \quad Y : g \Rightarrow X : f(g)$   
 $Y : g \quad X \setminus Y : f \Rightarrow X : f(g)$

Using  $\Lambda$ , UBL selects a logical form  $z$  for a sentence  $S$  by selecting the  $z$  with the most likely parse derivations  $y$ :  $h(S) = \arg \max_z \sum_y p(y, z | x; \theta, \Lambda)$ . The probabilistic model is a log-linear model with features for lexical entries used in the parse, as well as indicator features for relation-argument pairs in the logical form, to capture selectional preferences. Inference (parsing) and parameter estimation are driven by standard dynamic programming algorithms (Clark and Curran, 2007), while lexicon induction is based on a novel search procedure through the space of possible higher-order logic unification operations that yield the desired logical form for a training sentence.

Our Freebase data covers 81 of the 86 core domains in Freebase, and 635 of its over 2000 relations, but we wish to develop a semantic parser that can scale to all of Freebase. UBL gets us part of the way there, by inducing a PCCG grammar, as well as lexical entries for function words that must be handled in all domains. It can also learn lexical entries for relations  $r_D$  that appear in the training data. However, UBL has no way to learn lexical entries for the many valid  $(r_T, r_D)$  pairs that do not appear during training.

We use MATCHER’s learned alignment to extend the semantic parser that we get from UBL by automatically adding in lexical entries for Free-

base relations. Essentially, for each  $(r_T, r_D)$  from MATCHER’s output, we wish to construct a lexical entry that states that  $r_T$ ’s semantics resembles  $\lambda x \lambda y . r_D(x, y)$ . However, this simple process is complicated by the fact that the semantic parser requires two additional types of information for each lexical entry: a syntactic category, and a weight. Furthermore, for many cases the appropriate semantics are significantly more complex than this pattern.

To extend the learned semantic parser to a semantic parser for all of Freebase, we introduce a prediction task, which we call *semantic lexicon extension*: given a matching  $M$  together with scores for each pair in  $M$ , predict the syntactic category  $Syn$ , lambda-calculus semantics  $Sem$ , and weight  $W$  for a full lexical entry for each  $(r_T, r_D) \in M$ . One advantage of the reduction approach to learning a semantic parser is that we can automatically construct training examples for this prediction task from the other components in the reduction. We use the output lexical entries learned by UBL as (potentially noisy) examples of true lexical entries for  $(r_T, r_D)$  pairs where  $r_T$  matches the word in one of UBL’s lexical entries, and  $r_D$  forms part of the semantics in the same lexical entry. For  $(r_T, r_D)$  pairs in  $M$  where  $r_D$  occurs in UBL’s lexical entries, but not paired with  $r_T$ , we create dummy “negative” lexical entries with very low weights, one for each possible syntactic category observed in all lexical entries. Note that in order to train LEXTENDER, we need the output of MATCHER for the relations in UBL’s training data, as well as UBL’s output lexicon from the training data.

Our system for this prediction task, which we call LEXTENDER (for Lexicon eXtender), factors into three components:  $P(Sem | r_D, r_T, score)$ ,  $P(Syn | Sem, r_D, r_T, score)$ , and  $P(W | Syn, Sem, r_D, r_T, score)$ . This factorization is trivial in that it introduces no independence assumptions, but it helps in designing models for the task. We set the event space for random variable  $Sem$  to be the set of all lambda calculus expressions observed in UBL’s output lexicon, modulo the names of specific Freebase relations. For instance, if the lexicon includes two entries whose semantics are  $\lambda x \lambda y . \text{film\_actor}(x, y)$  and  $\lambda x \lambda y . \text{book\_author}(x, y)$ , the event space would include the single expression in which relations `film_actor` and `book_author` were replaced by

a new variable:  $\lambda p \lambda x \lambda y . p(x, y)$ . The final semantics for a lexical entry is then constructed by substituting  $r_D$  for  $p$ , or more formally, by a function application  $Sem(r_D)$ . The event space for  $Syn$  consists of all syntactic categories in UBL’s output lexicon, and  $W$  ranges over  $\mathbf{R}$ .

LEXTENDER’s model for  $Sem$  and  $Syn$  are Naïve Bayes classifiers (NBC), with features for the part-of-speech for  $r_T$  (taken from a POS tagger), the suffix of  $r_T$ , the number of arguments of  $r_D$ , and the argument types of  $r_D$ . For  $Syn$ , we add a feature for the predicted value of  $Sem$ . For  $W$ , we use a linear regression model whose features are the score from MATCHER, the probabilities from the  $Syn$  and  $Sem$  NBC models, and the average weight of all lexical entries in UBL with matching syntax and semantics. Using the predictions from these models, LEXTENDER extends UBL’s learned lexicon with all possible lexical entries with their predicted weights, although typically only a few lexical entries have high enough weight to make a difference during parsing. Pruning entries with low weights could improve the memory and time requirements for parsing, but these were not an issue in our experiments, so we did not investigate this further.

## 5 Experiments

We conducted experiments to test the ability of MATCHER and LEXTENDER to produce a semantic parser for Freebase. We first analyze MATCHER on the task of finding matches between Freebase relations and textual relations. We then compare the performance of the semantic parser learned by UBL with its extension provided by LEXTENDER on a dataset of English questions posed to Freebase.

### 5.1 Experimental Setup

Freebase (Bollacker et al., 2008) is a free, online, user-contributed, relational database (www.freebase.com) covering many different domains of knowledge. The full schema and contents are available for download. The “Freebase Commons” subset of Freebase, which is our focus, consists of 86 domains, an average of 25 relations per domain (total of 2134 relations), and 615,000 known instances per domain (53 million instances total). As a reference point, the GeoQuery database — which is a standard benchmark database for semantic parsing —

### Examples

1. What are the neighborhoods in New York City?  
 $\lambda x . \text{neighborhoods}(\text{new\_york}, x)$
2. How many countries use the rupee?  
 $\text{count}(x) . \text{countries\_used}(\text{rupee}, x)$
3. How many Peabody Award winners are there?  
 $\text{count}(x) . \exists y . \text{award\_honor}(y) \wedge$   
 $\text{award\_winner}(y, x) \wedge$   
 $\text{award}(y, \text{peabody\_award})$

Figure 2: Example questions with their logical forms. The logical forms make use of Freebase symbols as logical constants, as well as a few additional symbols such as `count` and `argmin`, to allow for aggregation queries.

contains a single domain (geography), 8 relations, and 880 total instances.

Our dataset contains 917 questions (on average, 6.3 words per question) and a meaning representation for each question written in a variant of lambda calculus<sup>2</sup>. 81 domains are represented in the data set, and the lambda calculus forms contain 635 distinct Freebase relations. The most common domains, `film` and `business`, each took up no more than 6% of the overall dataset. Several examples are listed in Fig. 2. The questions were provided by two native English speakers. No restrictions were placed on the type of questions they should produce, except that they should produce questions for multiple domains. By inspection, a large majority of the questions appear to be answerable from Freebase, although no instructions were given to restrict questions to this sort. We also created a dataset of alignments from these annotated questions by creating an alignment for each Freebase relation mentioned in the logical form for a question, paired with a manually-selected word from the question.

### 5.2 Alignment Tests

We measured the precision and recall of MATCHER’s output against the manually labeled data. Let  $M$  be the set of  $(r_T, r_D)$  matches produced by the system, and  $G$  the set of matches in the gold-standard manual data. We define

<sup>2</sup>The data is available from the second author’s website.

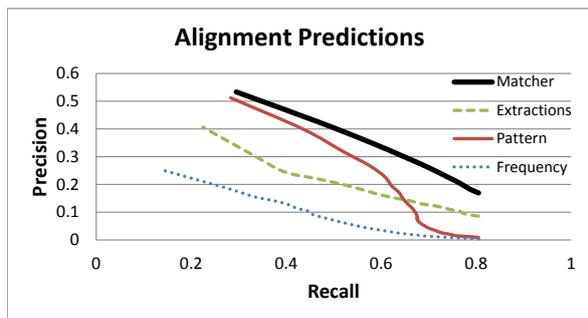


Figure 3: MATCHER’s Pattern features and Extractions features complement one another, so that in combination they outperform either subset on its own, especially at the high-recall end of the curve.

precision and recall as:

$$P = \frac{|M \cap G|}{|M|}, R = \frac{|M \cap G|}{|G|}$$

Figure 3 shows a Precision-Recall (PR) curve for MATCHER and three baselines: a “Frequency” model that ranks candidate matches for  $r_D$  by their frequency during the candidate identification step; a “Pattern” model that uses MATCHER’s linear regression model for ranking, but is restricted to only the pattern-based features; and an “Extractions” model that similarly restricts the ranking model to ReVerb features. We have three folds in our data; the alignments for relation  $r_D$  in one fold are predicted by models trained on the other two folds. Once all of the alignments in all three folds are scored, we generate points on the PR curve by applying a threshold to the model’s ranking, and treating all alignments above the threshold as the set of predicted alignments.

All regression models for learning alignments outperform the Frequency ranking by a wide margin. The Pattern model outperforms the Extractions model at the high-precision, low-recall end of the curve. At the high-recall points, the Pattern model drops quickly in precision. However, the combination of the two kinds of features in MATCHER yields improved precision at all levels of recall.

### 5.3 Semantic Parsing Tests

While our alignment tests can tell us in relative terms how well different models are performing, it is difficult to assess these models in absolute terms, since alignments are not typical applications that people care about in their own right. We

now compare our alignments on a semantic parsing task for Freebase.

In a first semantic parsing experiment, we train UBL, MATCHER, and LEXTENDER on a random sample of 70% of the questions, and test them on the remaining 30%. In a second test, we focus on the hard case where all questions from the test set contain logical constants that have never been seen before during training. We split the data into 3 folds, making sure that no Freebase domain has symbols appearing in questions in more than one fold. We then perform 3-fold cross-validation for all of our supervised models. We varied the number of matches that the alignment model (MATCHER, Pattern, Extractions, or Frequency) could make for each Freebase relation, and measured semantic parsing performance as a function of the number of matches.

Figure 4 shows the F1 scores for these semantic parsers, judged by exact match between the top-scoring logical form from the parser and the manually-produced logical form. Exact-match tests are overly-strict, in the sense that the system may be judged incorrect even when the logical form that is produced is logically equivalent to the correct logical form. However, by inspection such cases appear to be very rare in our data, and the exact-match criterion is often used in other semantic parsing experimental settings.

The semantic parsers produced by MATCHER+LEXTENDER and the other alignment techniques significantly outperform the baseline semantic parser learned by UBL, which achieves an overall F1 of 0.21 on these questions in the 70/30 split of the data, and an F1 of 0 in the cross-domain experiment. Purely-supervised approaches to this data are severely limited, since they have almost no chance of correctly parsing questions that refer to logical symbols that never appeared during training. However, MATCHER and LEXTENDER combine with UBL to produce an effective semantic parser. The best semantic parser we tested, which was produced by UBL, MATCHER, and LEXTENDER with 9 matches per Freebase relation, had a precision of 0.67 and a recall of 0.59 on the 70/30 split experiment.

The difference in alignment performance between MATCHER, Pattern, and Extractions carries over to semantic parsing. MATCHER drops in F1 with more matches as additional matches tend to be low-quality and low-probability, whereas Pat-

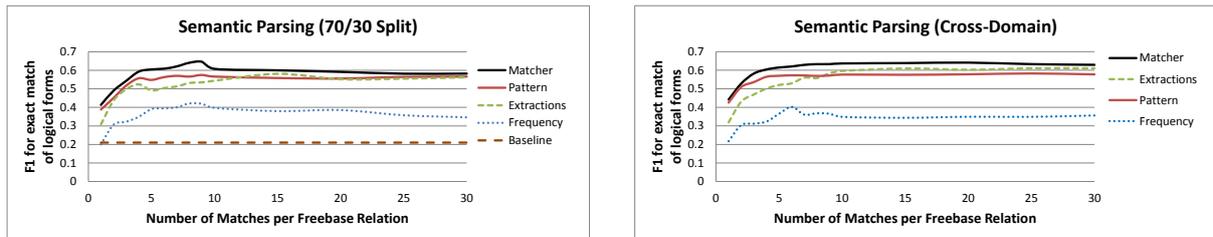


Figure 4: Semantic parsers produced by UBL+MATCHER+LEXTENDER outperform the purely-supervised baseline semantic parser on a random 70/30 split of the data (left) by as much as 0.42 in F1. In the case of this split and in the case of a cross-domain experiment (right), UBL+MATCHER+LEXTENDER outperforms UBL+Pattern+LEXTENDER by as much as 0.06 in F1.

tern and Extractions keep improving as more low-probability alignments are added. Interestingly, the Extractions model begins to overtake the Pattern model in F1 at higher numbers of matches, and all three models trend toward convergence in F1 with increasing numbers of matches. Nevertheless, MATCHER clearly improves over both, and reaches a higher F1 than either Pattern or Extractions using a small number of matches, which corresponds to a smaller lexicon and a leaner model.

To place these results in context, many different semantic parsers for databases like GeoQuery and ATIS (including parsers produced by UBL) have achieved F1 scores of 0.85 and higher. However, in all such tests, the test questions refer to logical constants that also appeared during training, allowing supervised techniques for learning semantic parsers to achieve strong accuracy. As we have argued, Freebase is large enough that is difficult to produce enough labeled training data to cover all of its logical constants. An unsupervised semantic parser for GeoQuery has achieved an F1 score of 0.66 (Goldwasser et al., 2011), impressive in its own right and slightly better than our F1 score. However, this parser was given questions which it knew *a priori* to contain words that refer to the logical constants in the database. Our MATCHER and LEXTENDER systems address a different challenge: how to learn a semantic parser for Freebase given the Web and a set of initial labeled questions.

## 6 Conclusion

Scaling semantic parsing to large databases requires an engineering effort to handle large datasets, but also novel algorithms to extend se-

matic parsing models to testing examples that look significantly different from labeled training data. The MATCHER and LEXTENDER algorithms represent an initial investigation into such techniques, with early results indicating that semantic parsers can handle Freebase questions on a large variety of domains with an F1 of 0.63.

We hope that our techniques and datasets will spur further research into this area. In particular, more research is needed to handle more complex matches between database and textual relations, and to handle more complex natural language queries. As mentioned in section 3.1, words like “actress” cannot be addressed by the current methodology, since MATCHER assumes that a word maps to a single Freebase relation, but the closest Freebase equivalent to the meaning of “actress” involves the two relations `film_actor` and `gender`. Another limitation is that our current methodology focuses on finding matches for nouns and verbs. Other important limitations of the current methodology include:

- the assumption that function words have no domain-specific meaning, which prepositions in particular can violate;
- low accuracy when there are few relevant results among the set of extracted relations;
- and the restriction to a single database (Freebase) for finding answers.

While significant challenges remain, the reduction of large-scale semantic parsing to a combination of schema matching and supervised learning offers a new path toward building high-coverage semantic parsers.

## References

- Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping Semantic Parsers from Conversations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. 2007. Open information extraction from the web. In *IJCAI*.
- Jacob Berlin and Amihai Motro. 2006. Database schema matching using machine learning with feature selection. In *Advanced Information Systems Engineering*. Springer.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, pages 1247–1250.
- Qingqing Cai and Alexander Yates. 2013. Semantic Parsing Freebase: Towards Open-Domain Semantic Parsing. In *Second Joint Conference on Lexical and Computational Semantics*.
- Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*.
- David L. Chen, Joohyun Kim, and Raymond J. Mooney. 2010. Training a Multilingual Sportscaster: Using Perceptual Context to Learn Language. *Journal of Artificial Intelligence Research*, 37:397–435.
- Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552.
- J. Clarke, D. Goldwasser, M. Chang, and D. Roth. 2010. Driving semantic parsing from the world’s response. In *Computational Natural Language Learning (CoNLL)*.
- R. Dhamanka, Y. Lee, A. Doan, A. Halevy, and P. Domingos. 2004. iMAP: Discovering Complex Semantic Matches between Database Schemas. In *SIGMOD*.
- A. Doan, J. Madhavan, P. Domingos, and A. Halevy. 2004. Ontology Matching: A Machine Learning Approach. In S. Staab and R. Studer, editors, *Handbook on Ontologies in Information Systems*, pages 397–416. Springer-Verlag.
- M. Ehrig, P. Haase, N. Stojanovic, and M. Hefke. 2004. Similarity for ontologies—a comprehensive framework. In *Workshop Enterprise Modelling and Ontology: Ingredients for Interoperability, PAKM*.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Fausto Giunchiglia, Pavel Shvaiko, and Mikalai Yatskevich. 2005. Semantic schema matching. *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, pages 347–365.
- D. Goldwasser, R. Reichart, J. Clarke, and D. Roth. 2011. Confidence driven unsupervised semantic parsing. In *Association for Computational Linguistics (ACL)*.
- Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. *Artificial Intelligence*, 194:28–61, January.
- Jayant Krishnamurthy and Tom Mitchell. 2012. Weakly Supervised Training of Semantic Parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing Probabilistic CCG Grammars from Logical Form with Higher-order Unification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- P. Liang, M. I. Jordan, and D. Klein. 2009. Learning semantic correspondences with less supervision. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.
- P. Liang, M. I. Jordan, and D. Klein. 2011. Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*.
- D. Lin and P. Pantel. 2001. DIRT – Discovery of Inference Rules from Text. In *KDD*.
- Henrik Nottelmann and Umberto Straccia. 2007. Information retrieval and machine learning for probabilistic schema matching. *Information processing & management*, 43(3):552–576.
- Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP ’09*, pages 1–10, Stroudsburg, PA, USA. Association for Computational Linguistics.
- E. Rahm and P.A. Bernstein. 2001. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10:334–350.
- P. D. Turney. 2001. Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In *Procs. of the Twelfth European Conference on Machine Learning (ECML)*, pages 491–502, Freiburg, Germany.

- M. Wick, K. Rohanimanesh, A. McCallum, and A.H. Doan. 2008a. A discriminative approach to ontology mapping. In *International Workshop on New Trends in Information Integration (NTII) at VLDB WS*.
- M.L. Wick, K. Rohanimanesh, K. Schultz, and A. McCallum. 2008b. A unified approach for schema matching, coreference and canonicalization. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*.
- Mohamed Yahya, Klaus Berberich, Shady Elbasuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural Language Questions for the Web of Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research (JAIR)*, 34:255–296, March.

# Fast and Accurate Shift-Reduce Constituent Parsing

Muhua Zhu<sup>†</sup>, Yue Zhang<sup>‡</sup>, Wenliang Chen<sup>\*</sup>, Min Zhang<sup>\*</sup> and Jingbo Zhu<sup>†</sup>

<sup>†</sup>Natural Language Processing Lab., Northeastern University, China

<sup>‡</sup>Singapore University of Technology and Design, Singapore

<sup>\*</sup> Soochow University, China and Institute for Infocomm Research, Singapore

zhumuhua@gmail.com

yue\_zhang@sutd.edu.sg

chenwenliang@gmail.com

mzhang@i2r.a-star.edu.sg

zhujingbo@mail.neu.edu.cn

## Abstract

Shift-reduce dependency parsers give comparable accuracies to their chart-based counterparts, yet the best shift-reduce constituent parsers still lag behind the state-of-the-art. One important reason is the existence of unary nodes in phrase structure trees, which leads to different numbers of shift-reduce actions between different outputs for the same input. This turns out to have a large empirical impact on the framework of global training and beam search. We propose a simple yet effective extension to the shift-reduce process, which eliminates size differences between action sequences in beam-search. Our parser gives comparable accuracies to the state-of-the-art chart parsers. With linear run-time complexity, our parser is over an order of magnitude faster than the fastest chart parser.

## 1 Introduction

Transition-based parsers employ a set of shift-reduce actions and perform parsing using a sequence of state transitions. The pioneering models rely on a classifier to make local decisions, and search greedily for a transition sequence to build a parse tree. Greedy, classifier-based parsers have been developed for both dependency grammars (Yamada and Matsumoto, 2003; Nivre et al., 2006) and phrase-structure grammars (Sagae and Lavie, 2005). With linear run-time complexity, they were commonly regarded as a faster but less accurate alternative to graph-based chart parsers (Collins, 1997; Charniak, 2000; McDonald et al., 2005).

Various methods have been proposed to address the disadvantages of greedy local parsing, among which a framework of beam-search and global discriminative training have been shown effective for dependency parsing (Zhang and Clark, 2008; Huang and Sagae, 2010). While beam-search reduces error propagation compared with greedy search, a discriminative model that is globally optimized for whole sequences of transition actions can avoid local score biases (Lafferty et al., 2001). This framework preserves the most important advantage of greedy local parsers, including linear run-time complexity and the freedom to define arbitrary features. With the use of rich non-local features, transition-based dependency parsers achieve state-of-the-art accuracies that are comparable to the best-graph-based parsers (Zhang and Nivre, 2011; Bohnet and Nivre, 2012). In addition, processing tens of sentences per second (Zhang and Nivre, 2011), these transition-based parsers can be a favorable choice for dependency parsing.

The above global-learning and beam-search framework can be applied to transition-based phrase-structure (constituent) parsing also (Zhang and Clark, 2009), maintaining all the aforementioned benefits. However, the effects were not as significant as for transition-based dependency parsing. The best reported accuracies of transition-based constituent parsers still lag behind the state-of-the-art (Sagae and Lavie, 2006; Zhang and Clark, 2009). One difference between phrase-structure parsing and dependency parsing is that for the former, parse trees with different numbers of unary rules require different numbers of actions to build. Hence the scoring model needs to disambiguate between transitions sequences with different sizes. For the same sentence, the largest output can take twice as many as actions to build as the

smallest one. This turns out to have a significant empirical impact on parsing with beam-search.

We propose an extension to the shift-reduce process to address this problem, which gives significant improvements to the parsing accuracies. Our method is conceptually simple, requiring only one additional transition action to eliminate size differences between different candidate outputs. On standard evaluations using both the Penn Treebank and the Penn Chinese Treebank, our parser gave higher accuracies than the Berkeley parser (Petrov and Klein, 2007), a state-of-the-art chart parser. In addition, our parser runs with over 89 sentences per second, which is 14 times faster than the Berkeley parser, and is the fastest that we are aware of for phrase-structure parsing. An open source release of our parser (version 0.6) is freely available on the Web.<sup>1</sup>

In addition to the above contributions, we apply a variety of semi-supervised learning techniques to our transition-based parser. These techniques have been shown useful to improve chart-based parsing (Koo et al., 2008; Chen et al., 2012), but little work has been done for transition-based parsers. We therefore fill a gap in the literature by reporting empirical results using these methods. Experimental results show that semi-supervised methods give a further improvement of 0.9% in F-score on the English data and 2.4% on the Chinese data. Our Chinese results are the best that we are aware of on the standard CTB data.

## 2 Baseline parser

We adopt the parser of Zhang and Clark (2009) for our baseline, which is based on the shift-reduce process of Sagae and Lavie (2005), and employs global perceptron training and beam search.

### 2.1 Vanilla Shift-Reduce

Shift-reduce parsing is based on a left-to-right scan of the input sentence. At each step, a transition action is applied to consume an input word or construct a new phrase-structure. A stack is used to maintain partially constructed phrase-structures, while the input words are stored in a buffer. The set of transition actions are

- *SHIFT*: pop the front word from the buffer, and push it onto the stack.

<sup>1</sup><http://sourceforge.net/projects/zpar/>

Axioms	$[\phi, 0, false, 0]$
Goal	$[S, n, true, C]$
Inference Rules:	
<i>SHIFT</i>	$\frac{[S, i, false, c]}{[S w, i + 1, false, c + c_s]}$
<i>REDUCE-L/R-X</i>	$\frac{[S s_1 s_0, i, false, c]}{[S X, i, false, c + c_r]}$
<i>UNARY-X</i>	$\frac{[S s_0, i, false, c]}{[S X, i, false, c + c_u]}$
<i>FINISH</i>	$\frac{[S, n, false, c]}{[S, n, true, c + c_f]}$

Figure 1: Deduction system of the baseline shift-reduce parsing process.

- *REDUCE-L/R-X*: pop the top two constituents off the stack, combine them into a new constituent with label X, and push the new constituent onto the stack.
- *UNARY-X*: pop the top constituent off the stack, raise it to a new constituent with label X, and push the new constituent onto the stack.
- *FINISH*: pop the root node off the stack and ends parsing.

The deduction system for the process is shown in Figure 1, where the item is formed as  $\langle stack, buffer\ front\ index, completion\ mark, score \rangle$ , and  $c_s$ ,  $c_r$ , and  $c_u$  represent the incremental score of the *SHIFT*, *REDUCE*, and *UNARY* parsing steps, respectively; these scores are calculated according to the context features of the parser state item.  $n$  is the number of words in the input.

### 2.2 Global Discriminative Training and Beam-Search

For a given input sentence, the initial state has an empty stack and a buffer that contains all the input words. An agenda is used to keep the  $k$  best state items at each step. At initialization, the agenda contains only the initial state. At each step, every state item in the agenda is popped and expanded by applying a valid transition action, and the top  $k$  from the newly constructed state items are put back onto the agenda. The process repeats until the agenda is empty, and the best completed state item (recorded as *candidate output*) is taken for

Description	Templates
unigrams	$s_0tc, s_0wc, s_1tc, s_1wc, s_2tc$ $s_2wc, s_3tc, s_3wc, q_0wt, q_1wt$ $q_2wt, q_3wt, s_0lwc, s_0rwc$ $s_0uwc, s_1lwc, s_1rwc, s_1uwc$
bigrams	$s_0ws_1w, s_0ws_1c, s_0cs_1w, s_0cs_1c,$ $s_0wq_0w, s_0wq_0t, s_0cq_0w, s_0cq_0t,$ $q_0wq_1w, q_0wq_1t, q_0tq_1w, q_0tq_1t,$ $s_1wq_0w, s_1wq_0t, s_1cq_0w, s_1cq_0t$
trigrams	$s_0cs_1cs_2c, s_0ws_1cs_2c, s_0cs_1wq_0t$ $s_0cs_1cs_2w, s_0cs_1cq_0t, s_0ws_1cq_0t$ $s_0cs_1wq_0t, s_0cs_1cq_0w$

Table 1: A summary of baseline feature templates, where  $s_i$  represents the  $i_{th}$  item on the stack  $S$  and  $q_i$  denotes the  $i_{th}$  item in the queue  $Q$ .  $w$  refers to the head lexicon,  $t$  refers to the head POS, and  $c$  refers to the constituent label.

the output.

The score of a state item is the total score of the transition actions that have been applied to build the item:

$$C(\alpha) = \sum_{i=1}^N \Phi(a_i) \cdot \vec{\theta}$$

Here  $\Phi(a_i)$  represents the feature vector for the  $i_{th}$  action  $a_i$  in state item  $\alpha$ . It is computed by applying the feature templates in Table 1 to the context of  $\alpha$ .  $N$  is the total number of actions in  $\alpha$ .

The model parameter  $\vec{\theta}$  is trained with the averaged perceptron algorithm, applied to state items (sequence of actions) globally. We apply the early update strategy (Collins and Roark, 2004), stopping parsing for parameter updates when the gold-standard state item falls off the agenda.

### 2.3 Baseline Features

Our baseline features are adopted from Zhang and Clark (2009), and are shown in Table 1 Here  $s_i$  represents the  $i_{th}$  item on the top of the stack  $S$  and  $q_i$  denotes the  $i_{th}$  item in the front end of the queue  $Q$ . The symbol  $w$  denotes the lexical head of an item; the symbol  $c$  denotes the constituent label of an item; the symbol  $t$  is the POS of a lexical head. These features are adapted from Zhang and Clark (2009). We remove Chinese specific features and make the baseline parser language-independent.

## 3 Improved hypotheses comparison

Unlike dependency parsing, constituent parse trees for the same sentence can have different numbers of nodes, mainly due to the existence of unary nodes. As a result, completed state

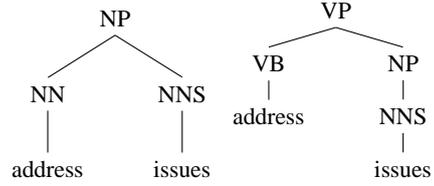


Figure 2: Example parse trees of the same sentence with different numbers of actions.

items for the same sentence can have different numbers of unary actions. Take the phrase “address issues” for example, two possible parses are shown in Figure 2 (a) and (b), respectively. The first parse corresponds to the action sequence [SHIFT, SHIFT, REDUCE-R-NP, FINISH], while the second parse corresponds to the action sequence [SHIFT, SHIFT, UNARY-NP, REDUCE-L-VP, FINISH], which consists of one more action than the first case. In practice, variances between state items can be much larger than the chosen example. In the extreme case where a state item does not contain any unary action, the number of actions is  $2n$ , where  $n$  is the number of words in the sentence. On the other hand, if the maximum number of consequent unary actions is 2 (Sagae and Lavie, 2005; Zhang and Clark, 2009), then the maximum number of actions a state item can have is  $4n$ .

The significant variance in the number of actions  $N$  can have an impact on the linear separability of state items, for which the feature vectors are  $\sum_{i=1}^N \Phi(a_i)$ . This turns out to have a significant empirical influence on perceptron training with early-update, where the training of the model interacts with search (Daume III, 2006).

One way of improving the comparability of state items is to reduce the differences in their sizes, and we use a *padding* method to achieve this. The idea is to extend the set of actions by adding an *IDLE* action, so that completed state items can be further expanded using the *IDLE* action. The action does not change the state itself, but simply adds to the number of actions in the sequence. A feature vector is extracted for the *IDLE* action according to the final state context, in the same way as other actions. Using the *IDLE* action, the transition sequence for the two parses in Figure 2 can be [SHIFT, SHIFT, REDUCE-NP, FINISH, IDLE] and [SHIFT, SHIFT, UNARY-NP, REDUCE-L-VP, FINISH], respectively. Their

Axioms	$[\phi, 0, \text{false}, 0, 0]$
Goal	$[S, n, \text{true}, m : 2n \leq m \leq 4n, C]$
	Inference Rules:
<i>SHIFT</i>	$\frac{[S, i, \text{false}, k, c]}{[S w, i + 1, \text{false}, k + 1, c + c_s]}$
<i>REDUCE-L/R-X</i>	$\frac{[S s_1 s_0, i, \text{false}, k, c]}{[S X, i, \text{false}, k + 1, c + c_r]}$
<i>UNARY-X</i>	$\frac{[S s_0, i, \text{false}, k, c]}{[S X, i, \text{false}, k + 1, c + c_u]}$
<i>FINISH</i>	$\frac{[S, n, \text{false}, k, c]}{[S, n, \text{true}, k + 1, c + c_f]}$
<i>IDLE</i>	$\frac{[S, n, \text{true}, k, c]}{[S, n, \text{true}, k + 1, c + c_i]}$

Figure 3: Deductive system of the extended transition system.

corresponding feature vectors have about the same sizes, and are more linearly separable. Note that there can be more than one action that are padded to a sequence of actions, and the number of IDLE actions depends on the size difference between the current action sequence and the largest action sequence without IDLE actions.

Given this extension, the deduction system is shown in Figure 3. We add the number of actions  $k$  to an item. The initial item (Axioms) has  $k = 0$ , while the goal item has  $2n \leq k \leq 4n$ . Given this process, beam-search decoding can be made simpler than that of Zhang and Clark (2009). While they used a *candidate output* to record the best completed state item, and finish decoding when the agenda contains no more items, we can simply finish decoding when all items in the agenda are completed, and output the best state item in the agenda. With this new transition process, we experimented with several extended features, and found that the templates in Table 2 are useful to improve the accuracies further. Here  $s_i ll$  denotes the left child of  $s_i$ 's left child. Other notations can be explained in a similar way.

#### 4 Semi-supervised Parsing with Large Data

This section discusses how to extract information from unlabeled data or auto-parsed data to further improve shift-reduce parsing accuracies. We consider three types of information, including

$s_0 llwc, s_0 lrwc, s_0 luwc$
$s_0 rlwc, s_0 rrwc, s_0 ruwc$
$s_0 ulwc, s_0 urwc, s_0 uuwc$
$s_1 llwc, s_1 lrwc, s_1 luwc$
$s_1 rlwc, s_1 rrwc, s_1 ruwc$

Table 2: New features for the extended parser.

paradigmatic relations, dependency relations, and structural relations. These relations are captured by word clustering, lexical dependencies, and a dependency language model, respectively. Based on the information, we propose a set of novel features specifically designed for shift-reduce constituent parsing.

#### 4.1 Paradigmatic Relations: Word Clustering

Word clusters are regarded as lexical intermediaries for dependency parsing (Koo et al., 2008) and POS tagging (Sun and Uszkoreit, 2012). We employ the Brown clustering algorithm (Liang, 2005) on unannotated data (word segmentation is performed if necessary). In the initial state of clustering, each word in the input corpus is regarded as a cluster, then the algorithm repeatedly merges pairs of clusters that cause the least decrease in the likelihood of the input corpus. The clustering results are a binary tree with words appearing as leaves. Each cluster is represented as a bit-string from the root to the tree node that represents the cluster. We define a function  $CLU(w)$  to return the cluster ID (a bit string) of an input word  $w$ .

#### 4.2 Dependency Relations: Lexical Dependencies

Lexical dependencies represent linguistic relations between words: whether a word modifies another word. The idea of exploiting lexical dependency information from auto-parsed data has been explored before for dependency parsing (Chen et al., 2009) and constituent parsing (Zhu et al., 2012).

To extract lexical dependencies, we first run the baseline parser on unlabeled data. To simplify the extraction process, we can convert auto-parsed constituency trees into dependency trees by using Penn2Malt.<sup>2</sup> From the dependency trees, we extract bigram lexical dependencies  $\langle w_1, w_2, L/R \rangle$  where the symbol  $L$  ( $R$ ) means that  $w_1$  ( $w_2$ ) is the head of  $w_2$  ( $w_1$ ). We also extract trigram lexical

<sup>2</sup><http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html>

dependencies  $\langle w_1, w_2, w_3, L/R \rangle$ , where  $L$  means that  $w_1$  is the head of  $w_2$  and  $w_3$ , meanwhile  $w_2$  and  $w_3$  are required to be siblings.

Following the strategy of Chen et al. (2009), we assign categories to bigram and trigram items **separately** according to their frequency counts. Specifically, top-10% most frequent items are assigned to the category of *High Frequency (HF)*; otherwise if an item is among top 20%, we assign it to the category of *Middle Frequency (MF)*; otherwise the category of *Low Frequency (LF)*. Hereafter, we refer to the bigram and trigram lexical dependency lists as *BLD* and *TLD*, respectively.

### 4.3 Structural Relations: Dependency Language Model

The dependency language model is proposed by Shen et al. (2008) and is used as additional information for graph-based dependency parsing in Chen et al. (2012). Formally, given a dependency tree  $y$  of an input sentence  $x$ , we can denote by  $H(y)$  the set of words that have at least one dependent. For each  $x_h \in H(y)$ , we have a corresponding dependency structure  $D_h = (x_{Lk}, \dots, x_{L1}, x_h, x_{R1}, \dots, x_{Rm})$ . The probability  $P(D_h)$  is defined to be

$$P(D_h) = P_L(D_h) \times P_R(D_h)$$

where  $P_L(D_h)$  can be in turn defined as:

$$\begin{aligned} P_L(D_h) \approx & P(x_{L1}|x_h) \\ & \times P(x_{L2}|x_{L1}, x_h) \\ & \times \dots \\ & \times P(x_{Lk}|x_{Lk-1}, \dots, x_{Lk-N+1}, x_h) \end{aligned}$$

$P_R(D_h)$  can be defined in a similar way.

We build dependency language models on auto-parsed data. Again, we convert constituency trees into dependency trees for the purpose of simplicity. From the dependency trees, we build a bigram and a trigram language model, which are denoted by BLM and TLM, respectively. The following are the templates of the records of the dependency language models.

(1)	$\langle x_{Li}, x_h, P(x_{Li} x_h) \rangle$
(2)	$\langle x_{Ri}, x_h, P(x_{Ri} x_h) \rangle$
(3)	$\langle x_{Li}, x_{Li-1}, x_h, P(x_{Li} x_{Li-1}, x_h) \rangle$
(4)	$\langle x_{Ri}, x_{Ri-1}, x_h, P(x_{Ri} x_{Ri-1}, x_h) \rangle$

Here the templates (1) and (2) belong to BLM and the templates (3) and (4) belong to TLM. To

	Stat	Train	Dev	Test	Unlabeled
EN	# sent	39.8k	1.7k	2.4k	3,139.1k
	# word	950.0k	40.1k	56.7k	76,041.4k
CH	# sent	18.1k	350	348	11,810.7k
	# word	493.8k	8.0k	6.8k	269,057.2k

Table 4: Statistics on sentence and word numbers of the experimental data.

use the dependency language models, we employ a map function  $\Phi(r)$  to assign a category to each record  $r$  according to its probability, as in Chen et al. (2012). The following is the map function.

$$\Phi(r) = \begin{cases} HP & \text{if } P(r) \in \text{top-10\%} \\ MP & \text{else if } P(r) \in \text{top-30\%} \\ LP & \text{otherwise} \end{cases}$$

### 4.4 Semi-supervised Features

We design a set of features based on the information extracted from auto-parsed data or unannotated data. The features are summarized in Table 3. Here *CLU* returns a cluster ID for a word. The functions  $BLD_{l/r}(\cdot)$ ,  $TLD_{l/r}(\cdot)$ ,  $BLM_{l/r}(\cdot)$ , and  $TLM_{l/r}(\cdot)$  check whether a given word combination can be found in the corresponding lists. For example,  $BLD_l(s_1w, s_0w)$  returns a category tag (*HF*, *MF*, or *LF*) if  $\langle s_1w, s_0w, L \rangle$  exists in the list BLD, else it returns *NONE*.

## 5 Experiments

### 5.1 Set-up

Labeled English data employed in this paper were derived from the Wall Street Journal (WSJ) corpus of the Penn Treebank (Marcus et al., 1993). We used sections 2-21 as labeled training data, section 24 for system development, and section 23 for final performance evaluation. For labeled Chinese data, we used the version 5.1 of the Penn Chinese Treebank (CTB) (Xue et al., 2005). Articles 001-270 and 440-1151 were used for training, articles 301-325 were used as development data, and articles 271-300 were used for evaluation.

For both English and Chinese data, we used ten-fold jackknifing (Collins, 2000) to automatically assign POS tags to the training data. We found that this simple technique could achieve an improvement of 0.4% on English and an improvement of 2.0% on Chinese. For English POS tagging, we adopted SVMTool,<sup>3</sup> and for Chinese POS tagging

<sup>3</sup><http://www.lsi.upc.edu/~nlp/SVMTool/>

Word Cluster Features		
$CLU(s_1w)$	$CLU(s_0w)$	$CLU(q_0w)$
$CLU(s_1w)s_1t$	$CLU(s_0w)s_0t$	$CLU(q_0w)q_0w$
Lexical Dependency Features		
$BLD_r(s_1w, s_0w)$	$BLD_l(s_1w, s_0w) \circ s_1t \circ s_0t$	$BLD_r(s_1w, s_0w)$
$BLD_r(s_1w, s_0w) \circ s_1t \circ s_0t$	$BLD_l(s_1w, q_0w) \circ s_1t \circ q_0t$	$BLD_l(s_1w, q_0w)$
$BLD_r(s_1w, q_0w)$	$BLD_r(s_1w, q_0w) \circ s_1t \circ q_0t$	$BLD_l(s_0w, q_0w)$
$BLD_l(s_0w, q_0w) \circ s_0t \circ q_0t$	$BLD_r(s_0w, q_0w) \circ s_0t \circ q_0t$	$BLD_r(s_0w, q_0w)$
$TLD_l(s_1w, s_1rdw, s_0w)$	$TLD_l(s_1w, s_1rdw, s_0w) \circ s_1t \circ s_0t$	$TLD_r(s_1w, s_0ldw, s_0w)$
$TLD_r(s_1w, s_0ldw, s_0w) \circ s_1t \circ s_0t$	$TLD_l(s_0w, s_0rdw, q_0w) \circ s_0t \circ q_0t$	$TLD_l(s_0w, s_0rdw, q_0w)$
$TLD_r(s_0w, NONE, q_0w)$	$TLD_r(s_0w, NONE, q_0w) \circ s_0t \circ q_0t$	
Dependency Language Model Features		
$BLM_l(s_1w, s_0w)$	$BLM_l(s_1w, s_0w) \circ s_1t \circ s_0t$	$BLM_r(s_1w, s_0w)$
$BLM_r(s_1w, s_0w) \circ s_1t \circ s_0t$	$BLM_l(s_0w, q_0w)$	$BLM_l(s_0w, q_0w) \circ s_0t \circ q_0t$
$BLM_r(s_0w, q_0w) \circ s_0t \circ q_0t$	$BLM_r(s_0w, q_0w)$	$TLM_l(s_1w, s_1rdw, s_0w)$
$TLM_l(s_1w, s_1rdw, s_0w) \circ s_1t \circ s_0t$	$TLM_r(s_1w, s_0ldw, s_0w)$	$TLM_r(s_1w, s_0ldw, s_0w) \circ s_1t \circ s_0t$

Table 3: Semi-supervised features designed on the base of word clusters, lexical dependencies, and dependency language models. Here the symbol  $s_i$  denotes a stack item,  $q_i$  denotes a queue item,  $w$  represents a word, and  $t$  represents a POS tag.

Lan.	System	LR	LP	F1
ENG	Baseline	88.4	88.7	88.6
	+padding	88.8	89.5	89.1
	+features	89.0	89.7	89.3
CHN	Baseline	85.6	86.3	86.0
	+padding	85.5	87.2	86.4
	+features	85.5	87.6	86.5

Table 5: Experimental results on the English and Chinese development sets with the padding technique and new supervised features added **incrementally**.

we employed the Stanford POS tagger.<sup>4</sup>

We took the WSJ articles from the TIPSTER corpus (LDC93T3A) as unlabeled English data. In addition, we removed from the unlabeled English data the sentences that appear in the WSJ corpus of the Penn Treebank. For unlabeled Chinese data, we used Chinese Gigaword (LDC2003T09), on which we conducted Chinese word segmentation by using a CRF-based segmenter. Table 4 summarizes data statistics on sentence and word numbers of the data sets listed above.

We used *EVALB* to evaluate parser performances, including labeled precision (LP), labeled recall (LR), and bracketing F1.<sup>5</sup> For significance tests, we employed the randomized permutation-based tool provided by Daniel Bikel.<sup>6</sup>

In both training and decoding, we set the beam size to 16, which achieves a good tradeoff between efficiency and accuracy. The optimal iteration number of perceptron learning is determined

<sup>4</sup><http://nlp.stanford.edu/software/tagger.shtml>

<sup>5</sup><http://nlp.cs.nyu.edu/evalb>

<sup>6</sup><http://www.cis.upenn.edu/~dbikel/software.html#comparator>

Lan.	Features	LR	LP	F1
ENG	+word cluster	89.3	90.0	89.7
	+lexical dependencies	89.7	90.3	90.0
	+dependency LM	90.0	90.6	90.3
CHN	+word cluster	85.7	87.5	86.6
	+lexical dependencies	87.2	88.6	87.9
	+dependency LM	87.2	88.7	88.0

Table 6: Experimental results on the English and Chinese development sets with different types of semi-supervised features added **incrementally** to the extended parser.

on the development sets. For word clustering, we set the cluster number to 50 for both the English and Chinese experiments.

## 5.2 Results on Development Sets

Table 5 reports the results of the extended parser (baseline + padding + supervised features) on the English and Chinese development sets. We integrated the padding method into the baseline parser, based on which we further incorporated the supervised features in Table 2. From the results we find that the padding method improves the parser accuracies by 0.5% and 0.4% on English and Chinese, respectively. Incorporating the supervised features in Table 2 gives further improvements of 0.2% on English and 0.1% on Chinese.

Based on the extended parser, we experimented different types of semi-supervised features by adding the features incrementally. The results are shown in Table 6. By comparing the results in Table 5 and the results in Table 6 we can see that the semi-supervised features achieve an overall improvement of 1.0% on the English data and an im-

Type	Parser	LR	LP	F1
SI	Ratnaparkhi (1997)	86.3	87.5	86.9
	Collins (1999)	88.1	88.3	88.2
	Charniak (2000)	89.5	89.9	89.5
	Sagae & Lavie (2005)*	86.1	86.0	86.0
	Sagae & Lavie (2006)*	87.8	88.1	87.9
	<b>Baseline</b>	<b>90.0</b>	<b>89.9</b>	<b>89.9</b>
	Petrov & Klein (2007)	90.1	90.2	90.1
	<b>Baseline+Padding</b>	<b>90.2</b>	<b>90.7</b>	<b>90.4</b>
	Carreras et al. (2008)	90.7	91.4	91.1
RE	Charniak & Johnson (2005)	91.2	91.8	91.5
	Huang (2008)	92.2	91.2	91.7
SE	Zhu et al. (2012)*	90.4	90.5	90.4
	<b>Baseline+Padding+Semi</b>	<b>91.1</b>	<b>91.5</b>	<b>91.3</b>
	Huang & Harper (2009)	91.1	91.6	91.3
	Huang et al. (2010) <sup>†</sup>	91.4	91.8	91.6
	McClosky et al. (2006)	92.1	92.5	92.3

Table 7: Comparison of our parsers and related work on the English test set. \* Shift-reduce parsers. <sup>†</sup> The results of self-training with a single latent annotation grammar.

Type	Parser	LR	LP	F1
SI	Charniak (2000)*	79.6	82.1	80.8
	Bikel (2004) <sup>†</sup>	79.3	82.0	80.6
	<b>Baseline</b>	<b>82.1</b>	<b>83.1</b>	<b>82.6</b>
	<b>Baseline+Padding</b>	<b>82.1</b>	<b>84.3</b>	<b>83.2</b>
	Petrov & Klein (2007)	81.9	84.8	83.3
RE	Charniak & Johnson (2005)*	80.8	83.8	82.3
SE	Zhu et al. (2012)	80.6	81.9	81.2
	<b>Baseline+Padding+Semi</b>	<b>84.4</b>	<b>86.8</b>	<b>85.6</b>

Table 8: Comparison of our parsers and related work on the test set of CTB5.1.\* Huang (2009) adapted the parsers to Chinese parsing on CTB5.1. <sup>†</sup> We run the parser on CTB5.1 to get the results.

provement of 1.5% on the Chinese data.

### 5.3 Final Results

Here we report the final results on the English and Chinese test sets. We compared the final results with a large body of related work. We grouped the parsers into three categories: single parsers (SI), discriminative reranking parsers (RE), and semi-supervised parsers (SE). Table 7 shows the comparative results on the English test set and Table 8 reports the comparison on the Chinese test set.

From the results we can see that our extended parser (baseline + padding + supervised features) outperforms the Berkeley parser by 0.3% on English, and is comparable with the Berkeley parser on Chinese (-0.1% less). Here *+padding* means the padding technique and the features in Table 2. After integrating semi-supervised features, the parsing accuracy on English is improved to 91.3%. We note that the performance is on the same level

Parser	#Sent/Second	
Ratnaparkhi (1997)	Unk	
Collins (1999)	3.5	
Charniak (2000)	5.7	
Sagae & Lavie (2005)*	3.7 <sup>‡</sup>	
Sagae & Lavie (2006) <sup>†</sup>	2.2 <sup>‡</sup>	
Petrov & Klein (2007)	6.2	
Carreras et al. (2008)	Unk	
	Baseline	100.7
This Paper	Baseline+Padding	89.5
	Baseline+Padding+Semi	46.8

Table 9: Comparison of running times on the English test set, where the time for loading models is excluded. \* The results of SVM-based shift-reduce parsing with greedy search. <sup>†</sup> The results of MaxEnt-based shift-reduce parser with best-first search. <sup>‡</sup> Times reported by authors running on different hardware.

as the performance of self-trained parsers, except for McClosky et al. (2006), which is based on the combination of reranking and self-training. On Chinese, the final parsing accuracy is 85.6%. To our knowledge, this is by far the best reported performance on this data set.

The padding technique, supervised features, and semi-supervised features achieve an overall improvement of 1.4% over the baseline on English, which is significant on the level of  $p < 10^{-5}$ . The overall improvement on Chinese is 3.0%, which is also significant on the level of  $p < 10^{-5}$ .

### 5.4 Comparison of Running Time

We also compared the running times of our parsers with the related single parsers. We ran timing tests on an Intel 2.3GHz processor with 8GB memory. The comparison is shown in Table 9. From the table, we can see that incorporating semi-supervised features decreases parsing speed, but the semi-supervised parser still has the advantage of efficiency over other parsers. Specifically, the semi-supervised parser is 7 times faster than the Berkeley parser. Note that Sagae & Lavie (2005) and Sagae & Lavie (2006) are also shift-reduce parsers, and their running times were evaluated on different hardwares. In practice, the running times of the shift-reduce parsers should be much shorter than the reported times in the table.

### 5.5 Error Analysis

We conducted error analysis for the three systems: the baseline parser, the extended parser with

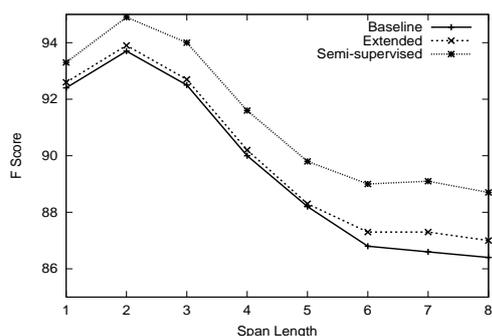


Figure 5: Comparison of parsing accuracies of the baseline, extended parser, and semi-supervised parsers on spans of different lengths.

the padding technique, and the semi-supervised parser, focusing on the English test set. The analysis was performed in four dimensions: parsing accuracies on different phrase types, on constituents of different span lengths, on different sentence lengths, and on sentences with different numbers of unknown words.

### 5.5.1 Different Phrase Types

Table 10 shows the parsing accuracies of the baseline, extended parser, and semi-supervised parser on different phrase types. Here we only consider the nine most frequent phrase types in the English test set. In the table, the phrase types are ordered from left to right in the descending order of their frequencies. We also show the improvements of the semi-supervised parser over the baseline parser (the last row in the table). As the results show, the extended parser achieves improvements on most of the phrase types with two exceptions: Preposition Phrase (PP) and Quantifier Phrase (QP). Semi-supervised features further improve parsing accuracies over the extended parser (QP is an exception). From the last row, we can see that improvements of the semi-supervised parser over the baseline on VP, S, SBAR, ADVP, and ADJP are above the average improvement (1.4%).

### 5.5.2 Different Span Lengths

Figure 5 shows a comparison of the three parsers on spans of different lengths. Here we consider span lengths up to 8. As the results show, both the padding extension and semi-supervised features are more helpful on relatively large spans: the performance gaps between the three parsers are enlarged with increasing span lengths.

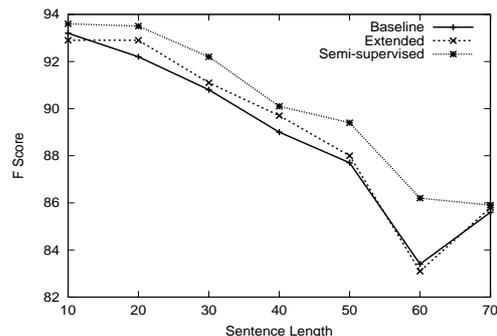


Figure 6: Comparison of parsing accuracies of the baseline, extended parser, and semi-supervised parser on sentences of different lengths.

### 5.5.3 Different Sentence Lengths

Figure 6 shows a comparison of parsing accuracies of the three parsers on sentences of different lengths. Each number on the horizontal axis represents the sentences whose lengths are between the number and its previous number. For example, the number 30 refers to the sentences whose lengths are between 20 and 30. From the results we can see that semi-supervised features improve parsing accuracy on both short and long sentences. The points at 70 are exceptions. In fact, sentences with lengths between 60 and 70 have only 8 instances, and the statistics on such a small number of sentences are not reliable.

### 5.5.4 Different Numbers of Unknown Words

Figure 4 shows a comparison of parsing accuracies of the baseline, extended parser, and semi-supervised parser on sentences with different numbers of unknown words. As the results show, the padding method is not very helpful on sentences with large numbers of unknown words, while semi-supervised features help significantly on this aspect. This conforms to the intuition that semi-supervised methods reduce data sparseness and improve the performance on unknown words.

## 6 Conclusion

In this paper, we addressed the problem of different action-sequence lengths for shift-reduce phrase-structure parsing, and designed a set of novel non-local features to further improve parsing. The resulting supervised parser outperforms the Berkeley parser, a state-of-the-art chart parser, in both accuracies and speeds. In addition, we incorporated a set of semi-supervised features. The

System	NP	VP	S	PP	SBAR	ADVP	ADJP	WHNP	QP
Baseline	91.9	90.1	89.8	88.1	85.7	84.6	72.1	94.8	89.3
Extended	92.1	90.7	90.2	87.9	86.6	84.5	73.6	95.5	88.6
Semi-supervised	93.2	92.0	91.5	89.3	88.2	86.8	75.1	95.7	89.1
Improvements	+1.3	+1.9	+1.7	+1.2	+2.5	+2.2	+3.0	+0.9	-0.2

Table 10: Comparison of parsing accuracies of the baseline, extended parser, and semi-supervised parsers on different phrase types.

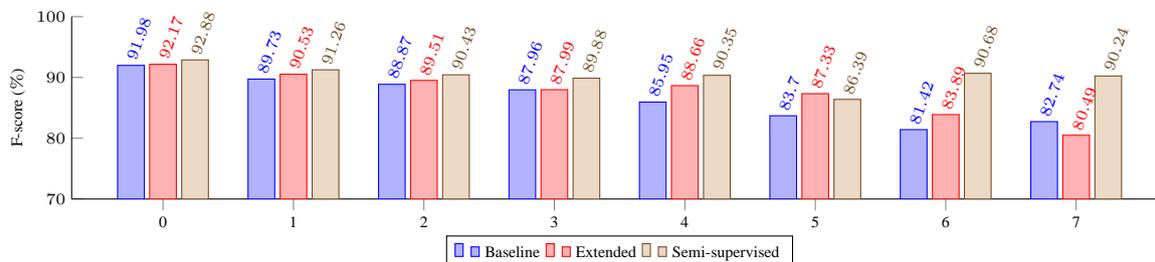


Figure 4: Comparison of parsing accuracies of the baseline, extended parser, and semi-supervised parser on sentences of different unknown words.

final parser reaches an accuracy of 91.3% on English and 85.6% on Chinese, by far the best reported accuracies on the CTB data.

## Acknowledgements

We thank the anonymous reviewers for their valuable comments. Yue Zhang and Muhua Zhu were supported partially by SRG-ISTD-2012-038 from Singapore University of Technology and Design. Muhua Zhu and Jingbo Zhu were funded in part by the National Science Foundation of China (61073140; 61272376), Specialized Research Fund for the Doctoral Program of Higher Education (20100042110031), and the Fundamental Research Funds for the Central Universities (N100204002). Wenliang Chen was funded partially by the National Science Foundation of China (61203314).

## References

- Daniel M. Bikel. 2004. *On the parameter space of generative lexicalized statistical parsing models*. Ph.D. thesis, University of Pennsylvania.
- Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of EMNLP*, pages 12–14, Jeju Island, Korea.
- Xavier Carreras, Michael Collins, and Terry Koo. 2008. Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of CoNLL*, pages 9–16, Manchester, England.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, pages 173–180.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL*, pages 132–139, Seattle, Washington, USA.
- Wenliang Chen, Junichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of EMNLP*, pages 570–579, Singapore.
- Wenliang Chen, Min Zhang, and Haizhou Li. 2012. Utilizing dependency language models for graph-based dependency. In *Proceedings of ACL*, pages 213–222, Jeju, Republic of Korea.
- Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of ACL*, Stroudsburg, PA, USA.
- Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL*, Madrid, Spain.
- Michael Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.
- Michael Collins. 2000. Discriminative reranking for natural language processing. In *Proceedings of ICML*, pages 175–182, Stanford, CA, USA.
- Hal Daume III. 2006. *Practical Structured Learning for Natural Language Processing*. Ph.D. thesis, USC.
- Zhongqiang Huang and Mary Harper. 2009. Self-training PCFG grammars with latent annotations

- across languages. In *Proceedings of EMNLP*, pages 832–841, Singapore.
- Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of ACL*, pages 1077–1086, Uppsala, Sweden.
- Zhongqiang Huang, Mary Harper, and Slav Petrov. 2010. Self-training with products of latent variable grammars. In *Proceedings of EMNLP*, pages 12–22, Massachusetts, USA.
- Liang Huang. 2008. Forest reranking: discriminative parsing with non-local features. In *Proceedings of ACL*, pages 586–594, Ohio, USA.
- Liang-Ya Huang. 2009. Improve Chinese parsing with Max-Ent reranking parser. In *Master Project Report*, Brown University.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, Massachusetts, USA, June.
- Percy Liang. 2005. Semi-supervised learning for natural language. Master’s thesis, Massachusetts Institute of Technology.
- Mitchell P. Marcus, Beatrice Santorini, and Mary A. Marcinkiewicz. 1993. Building a large annotated corpus of English. *Computational Linguistics*, 19(2):313–330.
- David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the HLT/NAACL, Main Conference*, pages 152–159, New York City, USA, June.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91–98, Ann Arbor, Michigan, June.
- Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. Maltparser: a data-driven parser-generator for dependency parsing. In *Proceedings of LREC*, pages 2216–2219.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HLT/NAACL*, pages 404–411, Rochester, New York, April.
- Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of EMNLP*, Rhode Island, USA.
- Kenji Sagae and Alon Lavie. 2005. A classifier-based parser with linear run-time complexity. In *Proceedings of IWPT*, pages 125–132, Vancouver, Canada.
- Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of HLT/NAACL, Companion Volume: Short Papers*, pages 129–132, New York, USA.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL*, pages 577–585, Ohio, USA.
- Weiwei Sun and Hans Uszkoreit. 2012. Capturing paradigmatic and syntagmatic lexical relations: towards accurate Chinese part-of-speech tagging. In *Proceedings of ACL*, Jeju, Republic of Korea.
- Nianwen Xue, Fei Xia, Fu dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*, pages 195–206, Nancy, France.
- Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL/HLT*, pages 888–896, Columbus, Ohio.
- Yue Zhang and Stephen Clark. 2009. Transition-based parsing of the Chinese Treebank using a global discriminative model. In *Proceedings of IWPT*, Paris, France, October.
- Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of ACL*, pages 188–193, Portland, Oregon, USA.
- Muhua Zhu, Jingbo Zhu, and Huizhen Wang. 2012. Exploiting lexical dependencies from large-scale data for better shift-reduce constituency parsing. In *Proceedings of COLING*, pages 3171–3186, Mumbai, India.

# Nonconvex Global Optimization for Latent-Variable Models\*

Matthew R. Gormley     Jason Eisner  
Department of Computer Science  
Johns Hopkins University, Baltimore, MD  
{mrg, jason}@cs.jhu.edu

## Abstract

Many models in NLP involve latent variables, such as unknown parses, tags, or alignments. Finding the optimal model parameters is then usually a difficult nonconvex optimization problem. The usual practice is to settle for *local* optimization methods such as EM or gradient ascent.

We explore how one might instead search for a *global* optimum in parameter space, using branch-and-bound. Our method would eventually find the global maximum (up to a user-specified  $\epsilon$ ) if run for long enough, but at any point can return a suboptimal solution together with an upper bound on the global maximum.

As an illustrative case, we study a generative model for dependency parsing. We search for the maximum-likelihood model parameters and corpus parse, subject to posterior constraints. We show how to formulate this as a mixed integer quadratic programming problem with nonlinear constraints. We use the Reformulation Linearization Technique to produce convex relaxations during branch-and-bound. Although these techniques do not yet provide a practical solution to our instance of this NP-hard problem, they sometimes find better solutions than Viterbi EM with random restarts, in the same time.

## 1 Introduction

Rich models with latent linguistic variables are popular in computational linguistics, but in general it is not known how to find their optimal parameters. In this paper, we present some “new” attacks for this common optimization setting, drawn from the mathematical programming toolbox.

We focus on the well-studied but unsolved task of unsupervised dependency parsing (i.e., depen-

\*This research was partially funded by the JHU Human Language Technology Center of Excellence.

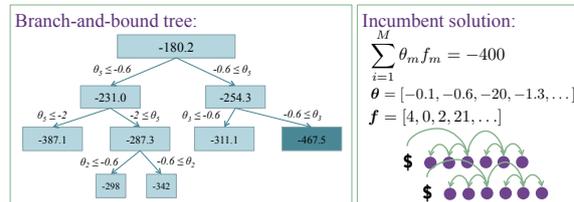


Figure 1: Each node contains a local upper bound for its subspace, computed by a relaxation. The node branches on a single model parameter  $\theta_m$  to partition its subspace. The lower bound, -400, is given by the best solution seen so far, the incumbent. The upper bound, -298, is the min of all remaining leaf nodes. The node with a local bound of -467.5 can be pruned because no solution within its subspace could be better than the incumbent.

dency grammar induction). This may be a particularly hard case, but its structure is typical. Many parameter estimation techniques have been attempted, including expectation-maximization (EM) (Klein and Manning, 2004; Spitzkovsky et al., 2010a), contrastive estimation (Smith and Eisner, 2006; Smith, 2006), Viterbi EM (Spitzkovsky et al., 2010b), and variational EM (Naseem et al., 2010; Cohen et al., 2009; Cohen and Smith, 2009). These are all *local search* techniques, which improve the parameters by hill-climbing.

The problem with local search is that it gets stuck in local optima. This is evident for grammar induction. An algorithm such as EM will find numerous different solutions when randomly initialized to different points (Charniak, 1993; Smith, 2006). A variety of ways to find better local optima have been explored, including heuristic initialization of the model parameters (Spitzkovsky et al., 2010a), random restarts (Smith, 2006), and annealing (Smith and Eisner, 2006; Smith, 2006). Others have achieved accuracy improvements by enforcing linguistically motivated *posterior constraints* on the parameters (Gillenwater et al., 2010; Naseem et al., 2010), such as requiring most sentences to have verbs or encouraging nouns to be children of verbs or prepositions.

We introduce a method that performs *global*

*search* with certificates of  $\epsilon$ -optimality for both the corpus parse and the model parameters. Our search objective is log-likelihood. We can also impose posterior constraints on the latent structure.

As we show, maximizing the joint log-likelihood of the parses and the parameters can be formulated as a mathematical program (MP) with a nonconvex quadratic objective and with integer linear and nonlinear constraints. Note that this objective is that of hard (Viterbi) EM—we do not marginalize over the parses as in classical EM.<sup>1</sup>

To globally optimize the objective function, we employ a branch-and-bound algorithm that searches the continuous space of the model parameters by branching on individual parameters (see Figure 1). Thus, our branch-and-bound tree serves to recursively subdivide the global parameter hypercube. Each node represents a search problem over one of the resulting boxes (i.e., orthotopes).

The crucial step is to prune nodes high in the tree by determining that their boxes *cannot* contain the global maximum. We compute an upper bound at each node by solving a relaxed maximization problem tailored to its box. If this upper bound is worse than our current best solution, we can prune the node. If not, we split the box again via another branching decision and retry on the two halves.

At each node, our relaxation derives a linear programming problem (LP) that can be efficiently solved by the dual simplex method. First, we linearly relax the constraints that grammar rule probabilities sum to 1—these constraints are nonlinear in our parameters, which are *log*-probabilities. Second, we linearize the quadratic objective by applying the Reformulation Linearization Technique (RLT) (Sherali and Adams, 1990), a method of forming tight linear relaxations of various types of MPs: the *reformulation* step multiplies together pairs of the original linear constraints to generate new quadratic constraints, and then the *linearization* step replaces quadratic terms in the new constraints with auxiliary variables.

Finally, if the node is not pruned, we search for a better incumbent solution under that node by projecting the solution of the RLT relaxation back onto the feasible region. In the relaxation, the model parameters might sum to slightly more than

<sup>1</sup>This objective might not be a great sacrifice: Spitzkovsky et al. (2010b) present evidence that hard EM can outperform soft EM for grammar induction in a hill-climbing setting. We use it because it is a quadratic objective. However, maximizing it remains NP-hard (Cohen and Smith, 2010).

one and the parses can consist of fractional dependency edges. We project in order to compute the true objective and compare with other solutions.

Our results demonstrate that our method can obtain higher likelihoods than Viterbi EM with random restarts. Furthermore, we show how posterior constraints inspired by Gillenwater et al. (2010) and Naseem et al. (2010) can easily be applied in our framework to obtain competitive accuracies using a simple model, the Dependency Model with Valence (Klein and Manning, 2004). We also obtain an  $\epsilon$ -optimal solution on a toy dataset.

We caution that the linear relaxations are very loose on larger boxes. Since we have many dimensions, the binary branch-and-bound tree may have to grow quite deep before the boxes become small enough to prune. This is why nonconvex quadratic optimization by LP-based branch-and-bound usually fails with more than 80 variables (Burer and Vandembussche, 2009). Even our smallest (toy) problems have hundreds of variables, so our experimental results mainly just illuminate the method’s behavior. Nonetheless, we offer the method as a new tool which, just as for local search, might be combined with other forms of problem-specific guidance to produce more practical results.

## 2 The Constrained Optimization Task

We begin by describing how for our typical model, the Viterbi EM objective can be formulated as a **mixed integer quadratic programming** (MIQP) problem with **nonlinear constraints** (Figure 2).

Other locally normalized log-linear generative models (Berg-Kirkpatrick et al., 2010) would have a similar formulation. In such models, the log-likelihood objective is simply a linear function of the feature counts. However, the objective becomes **quadratic** in unsupervised learning, because the feature counts are themselves unknown variables to be optimized. The feature counts are constrained to be derived from the latent variables (e.g., parses), which are unknown discrete structures that must be encoded with **integer** variables. The **nonlinear** constraints ensure that the model parameters are true log-probabilities.

Concretely, (1) specifies the Viterbi EM objective: the total log-probability of the *best* parse trees under the parameters  $\theta$ , given by a sum of log-probabilities  $\theta_m$  of the individual steps needed to generate the tree, as encoded by the features  $f_m$ . The (nonlinear) sum-to-one constraints on the

Variables:

$\theta_m$	Log-probability for feature $m$
$f_m$	Corpus-wide feature count for $m$
$e_{sij}$	Indicator of an arc from $i$ to $j$ in tree $s$

Indices and constants:

$m$	Feature / model parameter index
$s$	Sentence index
$c$	Conditional distribution index
$M$	Number of model parameters
$C$	Number of conditional distributions
$\mathcal{M}_c$	$c^{\text{th}}$ Set of feature indices that sum to 1.0
$S$	Number of sentences
$N_s$	Number of words in the $s^{\text{th}}$ sentence

Objective and constraints:

$\max \sum_m \theta_m f_m$	(1)
s.t. $\sum_{m \in \mathcal{M}_c} \exp(\theta_m) = 1, \forall c$	(2)
$A \begin{bmatrix} \mathbf{f} \\ \mathbf{e} \end{bmatrix} \leq b$	(Model constraints) (3)
$\theta_m \leq 0, f_m, e_{sij} \in \mathbb{Z}, \forall m, s, i, j$	(4)

Figure 2: Viterbi EM as a mathematical program

probabilities are in (2). The linear constraints in (3) will ensure that the arc variables for each sentence  $e_s$  encode a valid latent dependency tree, and that the  $\mathbf{f}$  variables count up the features of these trees. The final constraints (4) simply specify the range of possible values for the model parameters and their integer count variables.

Our experiments use the dependency model with valence (DMV) (Klein and Manning, 2004). This generative model defines a joint distribution over the sentences and their dependency trees.

We encode the DMV using integer linear constraints on the arc variables  $e$  and feature counts  $\mathbf{f}$ . These will constitute the model constraints in (3). The constraints must declaratively specify that the arcs form a valid dependency tree and that the resulting feature values are as defined by the DMV.

**Tree Constraints** To ensure that our arc variables,  $e_s$ , form a dependency tree, we employ the same single-commodity flow constraints of Magnanti and Wolsey (1994) as adapted by Martins et al. (2009) for parsing. We also use the projectivity constraints of Martins et al. (2009).

The single-commodity flow constraints simultaneously enforce that each node has exactly one parent, the special root node (position 0) has no in-

coming arcs, and the arcs form a connected graph.

For each sentence,  $s$ , the variable  $\phi_{sij}$  indicates the amount of flow traversing the arc from  $i$  to  $j$  in sentence  $s$ . The constraints below specify that the root node emits  $N_s$  units of flow (5), that one unit of flow is consumed by each each node (6), that the flow is zero on each disabled arc (7), and that the arcs are binary variables (8).

**Single-commodity flow** (Magnanti & Wolsey, 1994)

$$\sum_{j=1}^{N_s} \phi_{s0j} = N_s, \forall j \quad (5)$$

$$\sum_{i=0}^{N_s} \phi_{sij} - \sum_{k=1}^{N_s} \phi_{sjk} = 1, \forall j \quad (6)$$

$$\phi_{sij} \leq N_s e_{sij}, \forall i, j \quad (7)$$

$$e_{sij} \in \{0, 1\}, \forall i, j \quad (8)$$

Projectivity is enforced by adding a constraint (9) for each arc ensuring that no edges will cross that arc if it is enabled.  $\mathcal{X}_{ij}$  is the set of arcs  $(k, l)$  that cross the arc  $(i, j)$ .

**Projectivity** (Martins et al., 2009)

$$\sum_{(k,l) \in \mathcal{X}_{ij}} e_{skl} \leq N_s (1 - e_{sij}), \forall s, i, j \quad (9)$$

**DMV Feature Counts** The DMV generates a dependency tree recursively as follows. First the head word of the sentence is generated,  $t \sim \text{Discrete}(\boldsymbol{\theta}_{\text{root}})$ , where  $\boldsymbol{\theta}_{\text{root}}$  is a subvector of  $\boldsymbol{\theta}$ . To generate its children on the **left** side, we flip a coin to decide whether an adjacent child is generated,  $d \sim \text{Bernoulli}(\boldsymbol{\theta}_{\text{dec.L},0,t})$ . If the coin flip  $d$  comes up `continue`, we sample the word of that child as  $t' \sim \text{Discrete}(\boldsymbol{\theta}_{\text{child.L},t})$ . We continue generating non-adjacent children in this way, using coin weights  $\boldsymbol{\theta}_{\text{dec.L},\geq 1,t}$  until the coin comes up `stop`. We repeat this procedure to generate children on the **right** side, using the model parameters  $\boldsymbol{\theta}_{\text{dec.R},0,t}$ ,  $\boldsymbol{\theta}_{\text{child.R},t}$ , and  $\boldsymbol{\theta}_{\text{dec.R},\geq 1,t}$ . For each new child, we apply this process recursively to generate its descendants.

The feature count variables for the DMV are encoded in our MP as various sums over the edge variables. We begin with the **root/child feature counts**. The constraint (10) defines the feature count for model parameter  $\theta_{\text{root},t}$  as the number of all enabled arcs connecting the root node to a word of type  $t$ , summing over all sentences  $s$ . The constraint in (11) similarly defines  $f_{\text{child.L},t,t'}$  to be the number of enabled arcs connecting a parent of

type  $t$  to a left child of type  $t'$ .  $\mathcal{W}_{st}$  is the index set of tokens in sentences  $s$  with word type  $t$ .

DMV root/child feature counts	
$f_{\text{root},t} = \sum_{s=1}^{N_s} \sum_{j \in \mathcal{W}_{st}} e_{s0j}, \forall t$	(10)
$f_{\text{child.L},t,t'} = \sum_{s=1}^{N_s} \sum_{j < i} \delta \left[ \begin{smallmatrix} i \in \mathcal{W}_{st} \\ j \in \mathcal{W}_{st'} \end{smallmatrix} \right] e_{sij}, \forall t, t'$	(11)

The **decision feature counts** require the addition of an auxiliary count variables  $f_m^{(si)} \in \mathcal{Z}$  indicating how many times decision feature  $m$  fired at some position in the corpus  $s, i$ . We then need only add a constraint that the corpus wide feature count is the sum of these token-level feature counts  $f_m = \sum_{s=1}^S \sum_{i=1}^{N_s} f_m^{(si)}, \forall m$ .

Below we define these auxiliary variables for  $1 \leq s \leq S$  and  $1 \leq i \leq N_s$ . The helper variable  $n_{s,i,l}$  counts the number of enabled arcs to the left of token  $i$  in sentence  $s$ . Let  $t$  denote the word type of token  $i$  in sentence  $s$ . Constraints (11) - (16) are defined analogously for the *right side* feature counts.

DMV decision feature counts	
$n_{s,i,l} = \sum_{j=1}^{i-1} e_{sij}$	(12)
$n_{s,i,l}/N_s \leq f_{\text{dec.L},0,t,\text{cont}}^{(s,i)} \leq 1$	(13)
$f_{\text{dec.L},0,t,\text{stop}}^{(s,i)} = 1 - f_{\text{dec.L},0,t,\text{cont}}^{(s,i)}$	(14)
$f_{\text{dec.L},\geq 1,t,\text{stop}}^{(s,i)} = f_{\text{dec.L},0,t,\text{cont}}^{(s,i)}$	(15)
$f_{\text{dec.L},\geq 1,t,\text{cont}}^{(s,i)} = n_{s,i,l} - f_{\text{dec.L},0,t,\text{cont}}^{(s,i)}$	(16)

### 3 A Branch-and-Bound Algorithm

The mixed integer quadratic program with nonlinear constraints, given in the previous section, maximizes the nonconvex Viterbi EM objective and is NP-hard to solve (Cohen and Smith, 2010). The standard approach to optimizing this program is local search by the hard (Viterbi) EM algorithm. Yet local search can only provide a lower (pessimistic) bound on the global maximum.

We propose a branch-and-bound algorithm, which will iteratively tighten both pessimistic and optimistic bounds on the optimal solution. This algorithm may be halted at any time, to obtain the best current solution and a bound on how much better the global optimum could be.

A *feasible solution* is an assignment to all

the variables—both model parameters and corpus parse—that satisfies all constraints. Our branch-and-bound algorithm maintains an *incumbent solution*: the best known feasible solution according to the objective function. This is updated as better feasible solutions are found.

Our algorithm implicitly defines a search tree in which each node corresponds to a region of model parameter space. Our search procedure begins with only the root node, which represents the full model parameter space. At each node we perform three steps: bounding, projecting, and branching.

In the **bounding** step, we solve a relaxation of the original problem to provide an upper bound on the objective achievable within that node’s subregion. A node is pruned when  $L_{\text{global}} + \epsilon |L_{\text{global}}| \geq U_{\text{local}}$ , where  $L_{\text{global}}$  is the incumbent score,  $U_{\text{local}}$  is the upper bound for the node, and  $\epsilon > 0$ . This ensures that its entire subregion will not yield a  $\epsilon$ -better solution than the current incumbent.

The overall optimistic bound is given by the worst optimistic bound of all current leaf nodes.

The **projecting** step, if the node is not pruned, projects the solution of the relaxation back to the feasible region, replacing the current incumbent if this projection provides a better lower bound.

In the **branching** step, we choose a variable  $\theta_m$  on which to divide. Each of the child nodes receives a lower  $\theta_m^{\min}$  and upper  $\theta_m^{\max}$  bound for  $\theta_m$ . The child subspaces partition the parent subspace.

The search tree is defined by a variable ordering and the splitting procedure. We do binary branching on the variable  $\theta_m$  with the highest regret, defined as  $z_m - \theta_m f_m$ , where  $z_m$  is the auxiliary objective variable we will introduce in § 4.2. Since  $\theta_m$  is a log-probability, we split its current range at the midpoint in probability space,  $\log((\exp \theta_m^{\min} + \exp \theta_m^{\max})/2)$ .

We perform best-first search, ordering the nodes by the the optimistic bound of their parent. We also use the LP-guided rule (Martin, 2000; Achterberg, 2007, section 6.1) to perform depth-first plunges in search of better incumbents.

### 4 Relaxations

The relaxation in the bounding step computes an optimistic bound for a subspace of the model parameters. This upper bound would ideally be not much greater than the true maximum achievable on that region, but looser upper bounds are generally faster to compute.

We present successive relaxations to the original nonconvex mixed integer quadratic program with nonlinear constraints from (1)–(4). First, we show how the nonlinear sum-to-one constraints can be relaxed into linear constraints and tightened. Second, we apply a classic approach to bound the nonconvex quadratic objective by a linear concave envelope. Finally, we present our full relaxation based on the Reformulation Linearization Technique (RLT) (Sherali and Adams, 1990). We solve these LPs by the dual simplex algorithm.

#### 4.1 Relaxing the sum-to-one constraint

In this section, we use cutting planes to create a linear relaxation for the sum-to-one constraint (2). When relaxing a constraint, we must ensure that any assignment of the variables that was feasible (i.e. respected the constraints) in the original problem must also be feasible in the relaxation. In most cases, the relaxation is not perfectly tight and so will have an enlarged space of feasible solutions.

We begin by weakening constraint (2) to

$$\sum_{m \in \mathcal{M}_c} \exp(\theta_m) \leq 1 \quad (17)$$

The optimal solution under (17) still satisfies the original equality constraint (2) because of the maximization. We now relax (17) by approximating the surface  $z = \sum_{m \in \mathcal{M}_c} \exp(\theta_m)$  by the max of  $N$  lower-bounding linear functions on  $\mathbb{R}^{|\mathcal{M}_c|}$ . Instead of requiring  $z \leq 1$ , we only require each of these lower bounds to be  $\leq 1$ , slightly enlarging the feasible space into a convex polytope. Figure 3a shows the feasible region constructed from  $N=3$  linear functions on two log-probabilities  $\theta_1, \theta_2$ .

Formally, for each  $c$ , we define the  $i^{\text{th}}$  linear lower bound ( $i = 1, \dots, N$ ) to be the tangent hyperplane at some point  $\hat{\theta}_c^{(i)} = [\hat{\theta}_{c,1}^{(i)}, \dots, \hat{\theta}_{c,|\mathcal{M}_c|}^{(i)}] \in \mathbb{R}^{|\mathcal{M}_c|}$ , where each coordinate is a log-probability  $\hat{\theta}_{c,m}^{(i)} < 0$ . We require each of these linear functions to be  $\leq 1$ :

**Sum-to-one Relaxation**

$$\sum_{m \in \mathcal{M}_c} \left( \theta_m + 1 - \hat{\theta}_{c,m}^{(i)} \right) \exp \left( \hat{\theta}_{c,m}^{(i)} \right) \leq 1, \forall i, \forall c \quad (18)$$

#### 4.2 “Relaxing” the objective

Our true maximization objective  $\sum_m \theta_m f_m$  in (1) is a sum of quadratic terms. If the parameters  $\theta$

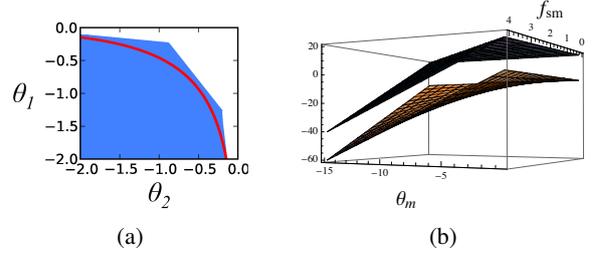


Figure 3: In (a), the area under the curve corresponds to those points  $(\theta_1, \theta_2)$  that satisfy (17) ( $z \leq 1$ ), with equality (2) achieved along the curve ( $z = 1$ ). The shaded area shows the enlarged feasible region under the linear relaxation. In (b), the curved lower surface represents a single product term in the objective. The piecewise-linear upper surface is its concave envelope (raised by 20 for illustration; in reality they touch).

were fixed, the objective would become linear in the latent features. Although the parameters are *not* fixed, the branch-and-bound algorithm does box them into a small region, where the quadratic objective is “more linear.”

Since it is easy to maximize a concave function, we will maximize the *concave envelope*—the concave function that most tightly upper-bounds our objective over the region. This turns out to be *piecewise linear* and can be maximized with an LP solver. Smaller regions yield tighter bounds.

Each node of the branch-and-bound tree specifies a region via bounds constraints  $\theta_m^{\min} < \theta_m < \theta_m^{\max}, \forall m$ . In addition, we have known bounds  $f_m^{\min} \leq f_m \leq f_m^{\max}, \forall m$  for the count variables.

McCormick (1976) described the concave envelope for a *single* quadratic term subject to bounds constraints (Figure 3b). In our case:

$$\theta_m f_m \leq \min \left[ f_m^{\max} \theta_m + \theta_m^{\min} f_m - \theta_m^{\min} f_m^{\max}, \right. \\ \left. f_m^{\min} \theta_m + \theta_m^{\max} f_m - \theta_m^{\max} f_m^{\min} \right]$$

We replace our objective  $\sum_m \theta_m f_m$  with  $\sum_m z_m$ , where we would like to constrain each auxiliary variable  $z_m$  to be  $= \theta_m f_m$  or (equivalently)  $\leq \theta_m f_m$ , but instead settle for making it  $\leq$  the concave envelope—a linear programming problem:

**Concave Envelope Objective**

$$\max \sum_m z_m \quad (19)$$

s.t.  $z_m \leq f_m^{\max} \theta_m + \theta_m^{\min} f_m - \theta_m^{\min} f_m^{\max}$  (20)

$z_m \leq f_m^{\min} \theta_m + \theta_m^{\max} f_m - \theta_m^{\max} f_m^{\min}$  (21)

### 4.3 Reformulation Linearization Technique

The *Reformulation Linearization Technique* (RLT)<sup>2</sup> (Sherali and Adams, 1990) is a method of forming tighter relaxations of various types of MPs. The basic method reformulates the problem by adding products of existing constraints. The quadratic terms in the objective and in these new constraints are redefined as auxiliary variables, thereby linearizing the program.

In this section, we will show how the RLT can be applied to our grammar induction problem and contrast it with the concave envelope relaxation presented in section 4.2.

Consider the original MP in equations (1) - (4), with the nonlinear sum-to-one constraints in (2) replaced by our linear constraints proposed in (18). If we remove the integer constraints in (4), the result is a quadratic program with *purely linear constraints*. Such problems have the form

$$\max x^T Qx \quad (22)$$

$$\text{s.t. } Ax \leq b \quad (23)$$

$$-\infty < L_i \leq x_i \leq U_i < \infty, \forall i \quad (24)$$

where the variables are  $x \in \mathbb{R}^n$ ,  $A$  is an  $m \times n$  matrix, and  $b \in \mathbb{R}^m$ , and  $Q$  is an  $n \times n$  indefinite<sup>3</sup> matrix. Without loss of generality we assume  $Q$  is symmetric. The application of the RLT here was first considered by Sherali and Tuncbilek (1995).

For convenience of presentation, we represent both the linear inequality constraints and the bounds constraints, under a different parameterization using the matrix  $G$  and vector  $g$ .

$$\begin{bmatrix} (b_i - A_i x) \geq 0, & 1 \leq i \leq m \\ (U_k - x_k) \geq 0, & 1 \leq k \leq n \\ (-L_k + x_k) \geq 0, & 1 \leq k \leq n \end{bmatrix} \equiv \begin{bmatrix} (g_i - G_i x) \geq 0, \\ 1 \leq i \leq m + 2n \end{bmatrix}$$

The *reformulation* step forms all possible products of these linear constraints and then adds them to the original quadratic program.

$$(g_i - G_i x)(g_j - G_j x) \geq 0, \forall 1 \leq i \leq j \leq m + 2n$$

In the *linearization* step, we replace all quadratic terms in the quadratic objective and new quadratic constraints with auxiliary variables:

$$w_{ij} \equiv x_i x_j, \forall 1 \leq i \leq j \leq n$$

<sup>2</sup>The key idea underlying the RLT was originally introduced in Adams and Sherali (1986) for 0-1 quadratic programming. It has since been extended to various other settings; see Sherali and Liberti (2008) for a complete survey.

<sup>3</sup>In the general case, that  $Q$  is indefinite causes this program to be nonconvex, making this problem NP-hard to solve (Vavasis, 1991; Pardalos, 1991).

This yields the following RLT relaxation:

**RLT Relaxation**

$$\max \sum_{1 \leq i \leq j \leq n} Q_{ij} w_{ij} \quad (25)$$

$$\text{s.t. } g_i g_j - \sum_{k=1}^n g_j G_{ik} x_k - \sum_{k=1}^n g_i G_{jk} x_k$$

$$+ \sum_{k=1}^n \sum_{l=1}^n G_{ik} G_{jl} w_{kl} \geq 0,$$

$$\forall 1 \leq i \leq j \leq m + 2n \quad (26)$$

Notice above that we have omitted the original inequality constraints (23) and bounds (24), because they are fully enforced by the new RLT constraints (26) from the reformulation step (Sherali and Tuncbilek, 1995). In our experiments, we keep the original constraints and instead explore subsets of the RLT constraints.

If the original QP contains equality constraints of the form  $G_e x = g_e$ , then we can form constraints by multiplying this one by each variable  $x_i$ . This gives us the following new set of constraints, for each equality constraint  $e$ :  $g_e x_i + \sum_{j=1}^n -G_{ej} w_{ij} = 0, \forall 1 \leq i \leq n$ .

**Theoretical Properties** The new constraints in eq. (26) will impose the concave envelope constraints (20)–(21) (Anstreicher, 2009).

The constraints presented above are considered to be *first-level* constraints corresponding to the first-level variables  $w_{ij}$ . However, the same technique can be applied repeatedly to produce polynomial constraints of higher degree. These higher level constraints/variables have been shown to provide increasingly tighter relaxations (Sherali and Adams, 1990) at the cost of a large number of variables and constraints. In the case where  $x \in \{0, 1\}^n$  the degree- $n$  RLT constraints will restrict to the convex hull of the feasible solutions (Sherali and Adams, 1990).

This is in direct contrast to the concave envelope relaxation presented in section 4.2 which relaxes to the convex hull of each quadratic term independently. This demonstrates the key intuition of the RLT relaxation: The products of constraints are implied (and unnecessary) in the original variable space. Yet when we project to a higher-dimensional space by including the auxiliary variables, the linearized constraints cut off portions of the feasible region given by only the concave envelope relaxation in eqs. (20)–(21).

#### 4.4 Adding Posterior Constraints

It is a simple extension to impose posterior constraints within our framework. Here we emphasize constraints that are analogous to the universal linguistic constraints from Naseem et al. (2010). Since we optimize the Viterbi EM objective, we directly constrain the counts in the single corpus parse rather than expected counts from a distribution over parses. Let  $\mathcal{E}$  be the index set of model parameters corresponding to edge types from Table 1 of Naseem et al. (2010), and  $N_s$  be the number of words in the  $s$ th sentence. We impose the constraint that 75% of edges come from  $\mathcal{E}$ :  $\sum_{m \in \mathcal{E}} f_m \geq 0.75 \left( \sum_{s=1}^S N_s \right)$ .

#### 5 Projections

A pessimistic bound, from the projecting step, will correspond to a feasible but not necessarily optimal solution to the original problem. We propose several methods for obtaining pessimistic bounds during the branch-and-bound search, by projecting and improving the solutions found by the relaxation. A solution to the relaxation may be infeasible in the original problem for two reasons: the model parameters might not sum to one, and/or the parse may contain fractional edges.

**Model Parameters** For each set of model parameters  $\mathcal{M}_c$  that should sum-to-one, we project the model parameters onto the  $\mathcal{M}_c - 1$  simplex by one of two methods: (1) normalize the infeasible parameters or (2) find the point on the simplex that has minimum Euclidean distance to the infeasible parameters using the algorithm of Chen and Ye (2011). For both methods, we can optionally apply add- $\lambda$  smoothing before projecting.

**Parses** Since we are interested in projecting the fractional parse onto the space of projective spanning trees, we can simply employ a dynamic programming parsing algorithm (Eisner and Satta, 1999) where the weight of each edge is given as the fraction of the edge variable.

Only one of these projection techniques is needed. We then either use *parsing* to fill in the optimal parse trees given the projected model parameters, or use *supervised parameter estimation* to fill in the optimal model parameters given the projected parses. These correspond to the Viterbi E step and M step, respectively. We can locally

improve the projected solution by continuing with a few additional iterations of Viterbi EM.

Related models could use very similar projection techniques. Given a relaxed joint solution to the parameters and the latent variables, one must be able to project it to a nearby feasible one, by projecting either the fractional parameters or the fractional latent variables into the feasible space and then solving exactly for the other.

#### 6 Related Work

The goal of this work was to better understand and address the non-convexity of maximum-likelihood training with latent variables, especially parses.

Gimpel and Smith (2012) proposed a concave model for unsupervised dependency parsing using IBM Model 1. This model did not include a tree constraint, but instead initialized EM on the DMV. By contrast, our approach incorporates the tree constraints directly into our convex relaxation and embeds the relaxation in a branch-and-bound algorithm capable of solving the original DMV maximum-likelihood estimation problem.

Spectral learning constitutes a wholly different family of consistent estimators, which achieve efficiency because they sidestep maximizing the nonconvex likelihood function. Hsu et al. (2009) introduced a spectral learner for a large class of HMMs. For supervised parsing, spectral learning has been used to learn latent variable PCFGs (Cohen et al., 2012) and hidden-state dependency grammars (Luque et al., 2012). Alas, there are not yet any spectral learning methods that recover latent tree *structure*, as in grammar induction.

Several integer linear programming (ILP) formulations of dependency parsing (Riedel and Clarke, 2006; Martins et al., 2009; Riedel et al., 2012) inspired our definition of grammar induction as a MP. Recent work uses branch-and-bound for decoding with non-local features (Qian and Liu, 2013). These differ from our work by treating the model parameters as constants, thereby yielding a linear objective.

For semi-supervised dependency parsing, Wang et al. (2008) used a convex objective, combining unsupervised least squares loss and a supervised large margin loss. This does not apply to our unsupervised setting. Branch-and-bound has also been applied to semi-supervised SVM training, a nonconvex search problem (Chapelle et al., 2007), with a relaxation derived from the dual.

## 7 Experiments

We first analyze the behavior of our method on a toy synthetic dataset. Next, we compare various parameter settings for branch-and-bound by estimating the total solution time. Finally, we compare our search method to Viterbi EM on a small subset of the Penn Treebank.

All our experiments use the DMV for unsupervised dependency parsing of part-of-speech (POS) tag sequences. For Viterbi EM we initialize the parameters of the model uniformly, breaking parser ties randomly in the first E-step (Spitkovsky et al., 2010b). This initializer is state-of-the-art for Viterbi EM. We also apply add-one smoothing during each M-step. We use random restarts, and select the model with the highest likelihood.

We add posterior constraints to Viterbi EM’s E-step. First, we run a relaxed linear programming (LP) parser, then project the (possibly fractional) parses back to the feasible region. If the resulting parse does not respect the posterior constraints, we discard it. The posterior constraint in the LP parser is tighter<sup>4</sup> than the one used in the true optimization problem, so the projections tends to be feasible under the true (looser) posterior constraints. In our experiments, all but one projection respected the constraints. We solve all LPs with CPLEX.

### 7.1 Synthetic Data

For our toy example, we generate sentences from a synthetic DMV over three POS tags (Verb, Noun, Adjective) with parameters chosen to favor short sentences with English word order.

In Figure 4 we show that the quality of the root relaxation increases as we approach the full set of RLT constraints. That the number of possible RLT constraints increases quadratically with the length of the corpus poses a serious challenge. For just 20 sentences from this synthetic model, the RLT generates 4,056,498 constraints.

For a single run of branch-and-bound, Figure 5 shows the global upper and lower bounds over time.<sup>5</sup> We consider five relaxations, each using only a subset of the RLT constraints. *Max.0k* uses only the concave envelope (20)-(21). *Max.1k* uses the concave envelope and also randomly samples 1,000 other RLT constraints, and so on for *Max.10k* and *Max.100k*. *Obj.Filter* includes all

<sup>4</sup>80% of edges must come from  $\mathcal{E}$  as opposed to 75%.

<sup>5</sup>The initial incumbent solution for branch-and-bound is obtained by running Viterbi EM with 10 random restarts.

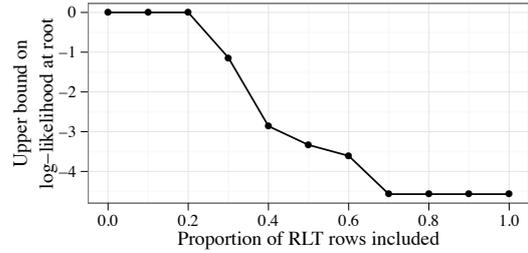


Figure 4: The bound quality at the root improves as the proportion of RLT constraints increases, on 5 synthetic sentences. A random subset of 70% of the 320,126 possible RLT constraints matches the relaxation quality of the full set. This bound is very tight: the relaxations in Figure 5 solve hundreds of nodes before such a bound is achieved.

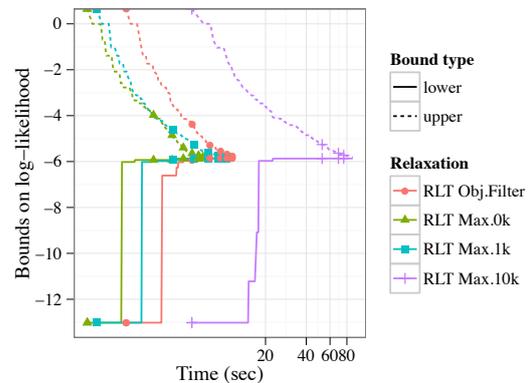


Figure 5: The global upper and lower bounds improve over time for branch-and-bound using different subsets of RLT constraints on 5 synthetic sentences. Each solves the problem to  $\epsilon$ -optimality for  $\epsilon = 0.01$ . A point marks every 200 nodes processed. (The time axis is log-scaled.)

constraints with a nonzero coefficient for one of the RLT variables  $z_m$  from the linearized objective. The rightmost lines correspond to *RLT Max.10k*: despite providing the tightest (local) bound at each node, it processed only 110 nodes in the time it took *RLT Max.1k* to process 1164. *RLT Max.0k* achieves the best balance of tight bounds and speed per node.

### 7.2 Comparing branch-and-bound strategies

It is prohibitively expensive to repeatedly run our algorithm to completion with a variety of parameter settings. Instead, we estimate the size of the branch-and-bound tree and the solution time using a high-variance estimate that is effective for comparisons (Lobjois and Lemaître, 1998).

Given a fixed set of parameters for our algorithm and an  $\epsilon$ -optimality stopping criterion, we

<i>RLT Relaxation</i>	<i>Avg. ms per node</i>	<i># Samples</i>	<i>Est. # Nodes</i>	<i>Est. # Hours</i>
Obj.Filter	63	10000	3.2E+08	4.6E+09
Max.0k	6	10000	1.7E+10	7.8E+10
Max.1k	15	10000	3.5E+08	4.2E+09
Max.10k	161	10000	1.3E+09	3.4E+10
Max.100k	232259	5	1.7E+09	9.7E+13

Table 1: Branch-and-bound node count and completion time estimates. Each standard deviation was close in magnitude to the estimate itself. We ran for 8 hours, stopping at 10,000 samples on 8 synthetic sentences.

can view the branch-and-bound tree  $T$  as fixed and finite in size. We wish to estimate some cost associated with the tree  $C(T) = \sum_{\alpha \in \text{nodes}(T)} f(\alpha)$ . Letting  $f(\alpha) = 1$  estimates the number of nodes; if  $f(\alpha)$  is the time to solve a node, then we estimate the total solution time using the Monte Carlo method of Knuth (1975). Table 1 gives these estimates, for the same five RLT relaxations. *Obj.Filter* yields the smallest estimated tree size.

### 7.3 Real Data

In this section, we compare our global search method to Viterbi EM with random restarts each with or without posterior constraints. We use 200 sentences of no more than 10 tokens from the WSJ portion of the Penn Treebank. We reduce the treebank’s gold part-of-speech (POS) tags to a universal set of 12 tags (Petrov et al., 2012) plus a tag for auxiliaries, ignoring punctuation. Each search method is run for 8 hours. We obtain the initial incumbent solution for branch-and-bound by running Viterbi EM for 45 minutes. The average time to solve a node’s relaxation ranges from 3 seconds for *RLT Max.0k* to 42 seconds for *RLT Max.100k*.

Figure 6a shows the log-likelihood of the incumbent solution over time. In our global search method, like Viterbi EM, the posterior constraints lead to lower log-likelihoods. *RLT Max.0k* finds the highest log-likelihood solution.

Figure 6b compares the *unlabeled directed dependency accuracy* of the incumbent solution. In both global and local search, the posterior constraints lead to higher accuracies. Viterbi EM with posterior constraints demonstrates the oscillation of incumbent accuracy: starting at 58.02% accuracy, it finds several high accuracy solutions early on (61.02%), but quickly abandons them to increase likelihood, yielding a final accuracy of 60.65%. *RLT Max.0k* with posterior constraints obtains the highest overall accuracy of 61.09% at

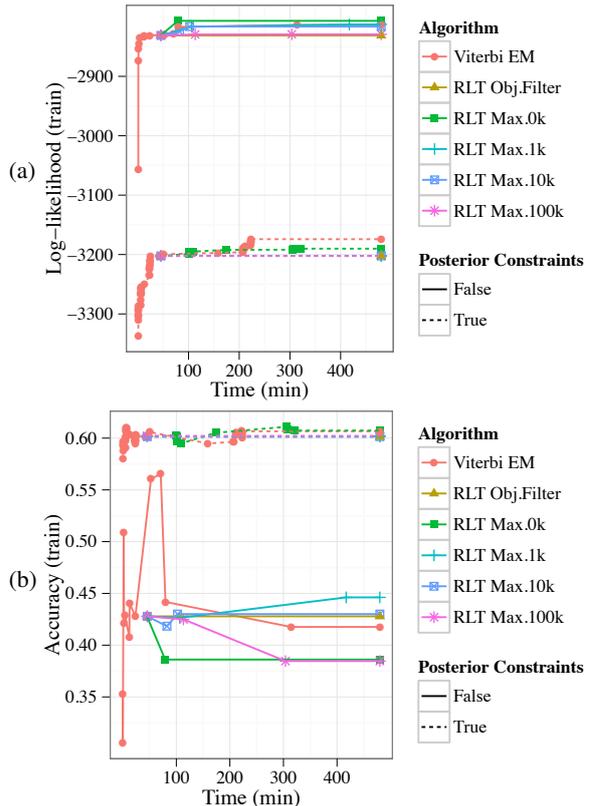


Figure 6: Likelihood (a) and accuracy (b) of incumbent solution so far, on a small real dataset.

306 min and the highest final accuracy 60.73%.

## 8 Discussion

In principle, our branch-and-bound method can approach  $\epsilon$ -optimal solutions to Viterbi training of locally normalized generative models, including the NP-hard case of grammar induction with the DMV. The method can also be used with posterior constraints or a regularized objective.

Future work includes algorithmic improvements for solving the relaxation and the development of tighter relaxations. The Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960) or Lagrangian Relaxation (Held and Karp, 1970) might satisfy both of these goals by pushing the integer tree constraints into a subproblem solved by a dynamic programming parser. Recent work on semidefinite relaxations (Anstreicher, 2009) suggests they may provide tighter bounds at the expense of greater computation time.

Perhaps even more important than tightening the bounds at each node are search heuristics (e.g., surface cues) and priors (e.g., universal grammar) that guide our global search by deciding *which* node to expand next (Chomsky and Lasnik, 1993).

## References

- Tobias Achterberg. 2007. *Constraint integer programming*. Ph.D. thesis, TU Berlin.
- Warren P. Adams and Hanif D. Sherali. 1986. A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, 32(10):1274–1290, October. ArticleType: research-article / Full publication date: Oct., 1986 / Copyright 1986 INFORMS.
- Kurt Anstreicher. 2009. Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. *Journal of Global Optimization*, 43(2):471–484.
- Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, DeNero, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Proc. of NAACL*, June.
- Samuel Burer and Dieter Vandenbussche. 2009. Globally solving box-constrained nonconvex quadratic programs with semidefinite-based finite branch-and-bound. *Computational Optimization and Applications*, 43(2):181–195.
- Olivier Chapelle, Vikas Sindhwani, and S. Sathya Keerthi. 2007. Branch and bound for semi-supervised support vector machines. In *Proc. of NIPS 19*, pages 217–224. MIT Press.
- E. Charniak. 1993. *Statistical language learning*. MIT press.
- Yunmei Chen and Xiaojing Ye. 2011. Projection onto a simplex. *arXiv:1101.6081*, January.
- Noam Chomsky and Howard Lasnik. 1993. Principles and parameters theory. In *Syntax: An International Handbook of Contemporary Research*. Berlin: de Gruyter.
- Shay Cohen and Noah A. Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proc. of HLT-NAACL*, pages 74–82, June.
- Shay Cohen and Noah A. Smith. 2010. Viterbi training for PCFGs: Hardness results and competitiveness of uniform initialization. In *Proc. of ACL*, pages 1502–1511, July.
- S. B. Cohen, K. Gimpel, and N. A. Smith. 2009. Logistic normal priors for unsupervised probabilistic grammar induction. In *Proceedings of NIPS*.
- Shay B. Cohen, Karl Stratos, Michael Collins, Dean P. Foster, and Lyle Ungar. 2012. Spectral learning of latent-variable PCFGs. In *Proc. of ACL (Volume 1: Long Papers)*, pages 223–231. Association for Computational Linguistics, July.
- George B. Dantzig and Philip Wolfe. 1960. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, January.
- Jason Eisner and Giorgio Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proc. of ACL*, pages 457–464, June.
- Jennifer Gillenwater, Kuzman Ganchev, Joo Graa, Fernando Pereira, and Ben Taskar. 2010. Sparsity in dependency grammar induction. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 194–199. Association for Computational Linguistics, July.
- K. Gimpel and N. A. Smith. 2012. Concavity and initialization for unsupervised dependency parsing. In *Proc. of NAACL*.
- M. Held and R. M. Karp. 1970. The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18(6):1138–1162.
- D. Hsu, S. M. Kakade, and T. Zhang. 2009. A spectral algorithm for learning hidden markov models. In *COLT 2009 - The 22nd Conference on Learning Theory*.
- Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*, pages 478–485, July.
- D. E. Knuth. 1975. Estimating the efficiency of backtrack programs. *Mathematics of computation*, 29(129):121–136.
- L. Lobjois and M. Lemaître. 1998. Branch and bound algorithm selection by performance prediction. In *Proc. of the National Conference on Artificial Intelligence*, pages 353–358.
- Franco M. Luque, Ariadna Quattoni, Borja Balle, and Xavier Carreras. 2012. Spectral learning for non-deterministic dependency parsing. In *Proc. of EACL*, pages 409–419, April.
- Thomas L. Magnanti and Laurence A. Wolsey. 1994. *Optimal Trees*. Center for Operations Research and Econometrics.
- Alexander Martin. 2000. Integer programs with block structure. Technical Report SC-99-03, ZIB.
- André Martins, Noah A. Smith, and Eric Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proc. of ACL-IJCNLP*, pages 342–350, August.
- Garth P. McCormick. 1976. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming*, 10(1):147–175.

- Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proc. of EMNLP*, pages 1234–1244, October.
- P. M. Pardalos. 1991. Global optimization algorithms for linearly constrained indefinite quadratic problems. *Computers & Mathematics with Applications*, 21(6):87–97.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC*.
- Xian Qian and Yang Liu. 2013. Branch and bound algorithm for dependency parsing with non-local features. *TACL*, 1:37–48.
- Sebastian Riedel and James Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proc. of EMNLP*, pages 129–137, July.
- Sebastian Riedel, David Smith, and Andrew McCallum. 2012. Parse, price and cut—Delayed column and row generation for graph based parsers. In *Proc. of EMNLP-CoNLL*, pages 732–743, July.
- Hanif D. Sherali and Warren P. Adams. 1990. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3(3):411–430, August.
- H. Sherali and L. Liberti. 2008. Reformulation-linearization technique for global optimization. *Encyclopedia of Optimization*, 2:3263–3268.
- Hanif D. Sherali and Cihan H. Tuncbilek. 1995. A reformulation-convexification approach for solving nonconvex quadratic programming problems. *Journal of Global Optimization*, 7(1):1–31.
- Noah A. Smith and Jason Eisner. 2006. Annealing structural bias in multilingual weighted grammar induction. In *Proc. of COLING-ACL*, pages 569–576, July.
- N.A. Smith. 2006. *Novel estimation methods for unsupervised discovery of latent structure in natural language text*. Ph.D. thesis, Johns Hopkins University, Baltimore, MD.
- Valentin I Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010a. From baby steps to leapfrog: How Less is more in unsupervised dependency parsing. In *Proc. of HLT-NAACL*, pages 751–759. Association for Computational Linguistics, June.
- Valentin I Spitkovsky, Hiyan Alshawi, Daniel Jurafsky, and Christopher D Manning. 2010b. Viterbi training improves unsupervised dependency parsing. In *Proc. of CoNLL*, pages 9–17. Association for Computational Linguistics, July.
- S. A. Vavasis. 1991. *Nonlinear optimization: complexity issues*. Oxford University Press, Inc.
- Qin Iris Wang, Dale Schuurmans, and Dekang Lin. 2008. Semi-supervised convex training for dependency parsing. In *Proc of ACL-HLT*, pages 532–540. Association for Computational Linguistics, June.

# Parsing with Compositional Vector Grammars

Richard Socher John Bauer Christopher D. Manning Andrew Y. Ng  
Computer Science Department, Stanford University, Stanford, CA 94305, USA

richard@socher.org, horatio@gmail.com, manning@stanford.edu, ang@cs.stanford.edu

## Abstract

Natural language parsing has typically been done with small sets of discrete categories such as NP and VP, but this representation does not capture the full syntactic nor semantic richness of linguistic phrases, and attempts to improve on this by lexicalizing phrases or splitting categories only partly address the problem at the cost of huge feature spaces and sparseness. Instead, we introduce a Compositional Vector Grammar (CVG), which combines PCFGs with a syntactically untied recursive neural network that learns syntactico-semantic, compositional vector representations. The CVG improves the PCFG of the Stanford Parser by 3.8% to obtain an F1 score of 90.4%. It is fast to train and implemented approximately as an efficient reranker it is about 20% faster than the current Stanford factored parser. The CVG learns a soft notion of head words and improves performance on the types of ambiguities that require semantic information such as PP attachments.

## 1 Introduction

Syntactic parsing is a central task in natural language processing because of its importance in mediating between linguistic expression and meaning. For example, much work has shown the usefulness of syntactic representations for subsequent tasks such as relation extraction, semantic role labeling (Gildea and Palmer, 2002) and paraphrase detection (Callison-Burch, 2008).

Syntactic descriptions standardly use coarse discrete categories such as NP for noun phrases or PP for prepositional phrases. However, recent work has shown that parsing results can be greatly improved by defining more fine-grained syntactic

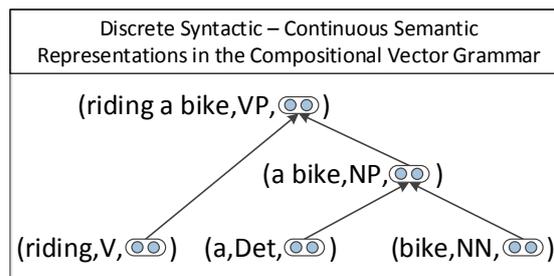


Figure 1: Example of a CVG tree with (category,vector) representations at each node. The vectors for nonterminals are computed via a new type of recursive neural network which is conditioned on syntactic categories from a PCFG.

categories, which better capture phrases with similar behavior, whether through manual feature engineering (Klein and Manning, 2003a) or automatic learning (Petrov et al., 2006). However, subdividing a category like NP into 30 or 60 subcategories can only provide a very limited representation of phrase meaning and semantic similarity. Two strands of work therefore attempt to go further. First, recent work in discriminative parsing has shown gains from careful engineering of features (Taskar et al., 2004; Finkel et al., 2008). Features in such parsers can be seen as defining effective dimensions of similarity between categories. Second, lexicalized parsers (Collins, 2003; Charniak, 2000) associate each category with a lexical item. This gives a fine-grained notion of semantic similarity, which is useful for tackling problems like ambiguous attachment decisions. However, this approach necessitates complex shrinkage estimation schemes to deal with the sparsity of observations of the lexicalized categories.

In many natural language systems, single words and  $n$ -grams are usefully described by their distributional similarities (Brown et al., 1992), among many others. But, even with large corpora, many

$n$ -grams will never be seen during training, especially when  $n$  is large. In these cases, one cannot simply use distributional similarities to represent unseen phrases. In this work, we present a new solution to learn features and phrase representations even for very long, unseen  $n$ -grams.

We introduce a Compositional Vector Grammar Parser (CVG) for structure prediction. Like the above work on parsing, the model addresses the problem of representing phrases and categories. Unlike them, it jointly learns how to parse and how to represent phrases as both discrete categories and continuous vectors as illustrated in Fig. 1. CVGs combine the advantages of standard probabilistic context free grammars (PCFG) with those of recursive neural networks (RNNs). The former can capture the discrete categorization of phrases into NP or PP while the latter can capture fine-grained syntactic and compositional-semantic information on phrases and words. This information can help in cases where syntactic ambiguity can only be resolved with semantic information, such as in the PP attachment of the two sentences: *They ate udon with forks.* vs. *They ate udon with chicken.*

Previous RNN-based parsers used the same (tied) weights at all nodes to compute the vector representing a constituent (Socher et al., 2011b). This requires the composition function to be extremely powerful, since it has to combine phrases with different syntactic head words, and it is hard to optimize since the parameters form a very deep neural network. We generalize the fully tied RNN to one with syntactically untied weights. The weights at each node are conditionally dependent on the categories of the child constituents. This allows different composition functions when combining different types of phrases and is shown to result in a large improvement in parsing accuracy.

Our compositional distributed representation allows a CVG parser to make accurate parsing decisions and capture similarities between phrases and sentences. Any PCFG-based parser can be improved with an RNN. We use a simplified version of the Stanford Parser (Klein and Manning, 2003a) as the base PCFG and improve its accuracy from 86.56 to 90.44% labeled F1 on all sentences of the WSJ section 23. The code of our parser is available at [nlp.stanford.edu](http://nlp.stanford.edu).

## 2 Related Work

The CVG is inspired by two lines of research: Enriching PCFG parsers through more diverse

sets of discrete states and recursive deep learning models that jointly learn classifiers and continuous feature representations for variable-sized inputs.

### Improving Discrete Syntactic Representations

As mentioned in the introduction, there are several approaches to improving discrete representations for parsing. Klein and Manning (2003a) use manual feature engineering, while Petrov et al. (2006) use a learning algorithm that splits and merges the syntactic categories in order to maximize likelihood on the treebank. Their approach splits categories into several dozen subcategories. Another approach is lexicalized parsers (Collins, 2003; Charniak, 2000) that describe each category with a lexical item, usually the head word. More recently, Hall and Klein (2012) combine several such annotation schemes in a factored parser. We extend the above ideas from discrete representations to richer continuous ones. The CVG can be seen as factoring discrete and continuous parsing in one model. Another different approach to the above generative models is to learn discriminative parsers using many well designed features (Taskar et al., 2004; Finkel et al., 2008). We also borrow ideas from this line of research in that our parser combines the generative PCFG model with discriminatively learned RNNs.

### Deep Learning and Recursive Deep Learning

Early attempts at using neural networks to describe phrases include Elman (1991), who used recurrent neural networks to create representations of sentences from a simple toy grammar and to analyze the linguistic expressiveness of the resulting representations. Words were represented as one-on vectors, which was feasible since the grammar only included a handful of words. Collobert and Weston (2008) showed that neural networks can perform well on sequence labeling language processing tasks while also learning appropriate features. However, their model is lacking in that it cannot represent the recursive structure inherent in natural language. They partially circumvent this problem by using either independent window-based classifiers or a convolutional layer. RNN-specific training was introduced by Goller and Küchler (1996) to learn distributed representations of given, structured objects such as logical terms. In contrast, our model both predicts the structure and its representation.

Henderson (2003) was the first to show that neural networks can be successfully used for large scale parsing. He introduced a left-corner parser to estimate the probabilities of parsing decisions conditioned on the parsing history. The input to Henderson’s model consists of pairs of frequent words and their part-of-speech (POS) tags. Both the original parsing system and its probabilistic interpretation (Titov and Henderson, 2007) learn features that represent the *parsing history* and do not provide a principled linguistic representation like our phrase representations. Other related work includes (Henderson, 2004), who discriminatively trains a parser based on synchrony networks and (Titov and Henderson, 2006), who use an SVM to adapt a generative parser to different domains.

Costa et al. (2003) apply recursive neural networks to re-rank possible phrase attachments in an incremental parser. Their work is the first to show that RNNs can capture enough information to make correct parsing decisions, but they only test on a subset of 2000 sentences. Menchetti et al. (2005) use RNNs to re-rank different parses. For their results on full sentence parsing, they re-rank candidate trees created by the Collins parser (Collins, 2003). Similar to their work, we use the idea of letting discrete categories reduce the search space during inference. We compare to fully tied RNNs in which the same weights are used at every node. Our syntactically untied RNNs outperform them by a significant margin. The idea of untying has also been successfully used in deep learning applied to vision (Le et al., 2010).

This paper uses several ideas of (Socher et al., 2011b). The main differences are (i) the dual representation of nodes as discrete categories and vectors, (ii) the combination with a PCFG, and (iii) the syntactic untying of weights based on child categories. We directly compare models with fully tied and untied weights. Another work that represents phrases with a dual discrete-continuous representation is (Kartsaklis et al., 2012).

### 3 Compositional Vector Grammars

This section introduces Compositional Vector Grammars (CVGs), a model to jointly find syntactic structure and capture compositional semantic information.

CVGs build on two observations. Firstly, that a lot of the structure and regularity in languages can be captured by well-designed syntactic patterns.

Hence, the CVG builds on top of a standard PCFG parser. However, many parsing decisions show fine-grained semantic factors at work. Therefore we combine syntactic and semantic information by giving the parser access to rich syntactico-semantic information in the form of distributional word vectors and compute compositional semantic vector representations for longer phrases (Costa et al., 2003; Menchetti et al., 2005; Socher et al., 2011b). The CVG model merges ideas from both generative models that assume discrete syntactic categories and discriminative models that are trained using continuous vectors.

We will first briefly introduce single word vector representations and then describe the CVG objective function, tree scoring and inference.

#### 3.1 Word Vector Representations

In most systems that use a vector representation for words, such vectors are based on co-occurrence statistics of each word and its context (Turney and Pantel, 2010). Another line of research to learn distributional word vectors is based on neural language models (Bengio et al., 2003) which jointly learn an embedding of words into an  $n$ -dimensional feature space and use these embeddings to predict how suitable a word is in its context. These vector representations capture interesting linear relationships (up to some accuracy), such as  $king - man + woman \approx queen$  (Mikolov et al., 2013).

Collobert and Weston (2008) introduced a new model to compute such an embedding. The idea is to construct a neural network that outputs high scores for windows that occur in a large unlabeled corpus and low scores for windows where one word is replaced by a random word. When such a network is optimized via gradient ascent the derivatives backpropagate into the word embedding matrix  $X$ . In order to predict correct scores the vectors in the matrix capture co-occurrence statistics.

For further details and evaluations of these embeddings, see (Turian et al., 2010; Huang et al., 2012). The resulting  $X$  matrix is used as follows. Assume we are given a sentence as an ordered list of  $m$  words. Each word  $w$  has an index  $[w] = i$  into the columns of the embedding matrix. This index is used to retrieve the word’s vector representation  $a_w$  using a simple multiplication with a binary vector  $e$ , which is zero everywhere, except

at the  $i$ th index. So  $a_w = Le_i \in \mathbb{R}^n$ . Henceforth, after mapping each word to its vector, we represent a sentence  $S$  as an ordered list of (word,vector) pairs:  $x = ((w_1, a_{w_1}), \dots, (w_m, a_{w_m}))$ .

Now that we have discrete and continuous representations for all words, we can continue with the approach for computing tree structures and vectors for nonterminal nodes.

### 3.2 Max-Margin Training Objective for CVGs

The goal of supervised parsing is to learn a function  $g : \mathcal{X} \rightarrow \mathcal{Y}$ , where  $\mathcal{X}$  is the set of sentences and  $\mathcal{Y}$  is the set of all possible labeled binary parse trees. The set of all possible trees for a given sentence  $x_i$  is defined as  $Y(x_i)$  and the correct tree for a sentence is  $y_i$ .

We first define a structured margin loss  $\Delta(y_i, \hat{y})$  for predicting a tree  $\hat{y}$  for a given correct tree. The loss increases the more incorrect the proposed parse tree is (Goodman, 1998). The discrepancy between trees is measured by counting the number of nodes  $N(y)$  with an incorrect span (or label) in the proposed tree:

$$\Delta(y_i, \hat{y}) = \sum_{d \in N(\hat{y})} \kappa \mathbf{1}\{d \notin N(y_i)\}. \quad (1)$$

We set  $\kappa = 0.1$  in all experiments. For a given set of training instances  $(x_i, y_i)$ , we search for the function  $g_\theta$ , parameterized by  $\theta$ , with the smallest expected loss on a new sentence. It has the following form:

$$g_\theta(x) = \arg \max_{\hat{y} \in Y(x)} s(\text{CVG}(\theta, x, \hat{y})), \quad (2)$$

where the tree is found by the Compositional Vector Grammar (CVG) introduced below and then scored via the function  $s$ . The higher the score of a tree the more confident the algorithm is that its structure is correct. This max-margin, structure-prediction objective (Taskar et al., 2004; Ratliff et al., 2007; Socher et al., 2011b) trains the CVG so that the highest scoring tree will be the correct tree:  $g_\theta(x_i) = y_i$  and its score will be larger up to a margin to other possible trees  $\hat{y} \in \mathcal{Y}(x_i)$ :

$$s(\text{CVG}(\theta, x_i, y_i)) \geq s(\text{CVG}(\theta, x_i, \hat{y})) + \Delta(y_i, \hat{y}).$$

This leads to the regularized risk function for  $m$

training examples:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m r_i(\theta) + \frac{\lambda}{2} \|\theta\|_2^2, \text{ where}$$

$$r_i(\theta) = \max_{\hat{y} \in Y(x_i)} (s(\text{CVG}(x_i, \hat{y})) + \Delta(y_i, \hat{y})) - s(\text{CVG}(x_i, y_i)) \quad (3)$$

Intuitively, to minimize this objective, the score of the correct tree  $y_i$  is increased and the score of the highest scoring incorrect tree  $\hat{y}$  is decreased.

### 3.3 Scoring Trees with CVGs

For ease of exposition, we first describe how to score an existing fully labeled tree with a standard RNN and then with a CVG. The subsequent section will then describe a bottom-up beam search and its approximation for finding the optimal tree.

Assume, for now, we are given a labeled parse tree as shown in Fig. 2. We define the word representations as (vector, POS) pairs:  $((a, A), (b, B), (c, C))$ , where the vectors are defined as in Sec. 3.1 and the POS tags come from a PCFG. The standard RNN essentially ignores all POS tags and syntactic categories and each non-terminal node is associated with the same neural network (i.e., the weights across nodes are fully tied). We can represent the binary tree in Fig. 2 in the form of branching triplets  $(p \rightarrow c_1 c_2)$ . Each such triplet denotes that a parent node  $p$  has two children and each  $c_k$  can be either a word vector or a non-terminal node in the tree. For the example in Fig. 2, we would get the triples  $((p^1 \rightarrow bc), (p^2 \rightarrow ap^1))$ . Note that in order to replicate the neural network and compute node representations in a bottom up fashion, the parent must have the same dimensionality as the children:  $p \in \mathbb{R}^n$ .

Given this tree structure, we can now compute activations for each node from the bottom up. We begin by computing the activation for  $p^1$  using the children's word vectors. We first concatenate the children's representations  $b, c \in \mathbb{R}^{n \times 1}$  into a vector  $\begin{bmatrix} b \\ c \end{bmatrix} \in \mathbb{R}^{2n \times 1}$ . Then the composition function multiplies this vector by the parameter weights of the RNN  $W \in \mathbb{R}^{n \times 2n}$  and applies an element-wise nonlinearity function  $f = \tanh$  to the output vector. The resulting output  $p^{(1)}$  is then given as input to compute  $p^{(2)}$ .

$$p^{(1)} = f \left( W \begin{bmatrix} b \\ c \end{bmatrix} \right), p^{(2)} = f \left( W \begin{bmatrix} a \\ p^{(1)} \end{bmatrix} \right)$$

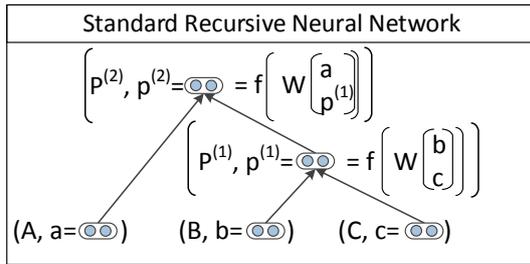


Figure 2: An example tree with a simple Recursive Neural Network: The same weight matrix is replicated and used to compute all non-terminal node representations. Leaf nodes are  $n$ -dimensional vector representations of words.

In order to compute a score of how plausible of a syntactic constituent a parent is the RNN uses a single-unit linear layer for all  $i$ :

$$s(p^{(i)}) = v^T p^{(i)},$$

where  $v \in \mathbb{R}^n$  is a vector of parameters that need to be trained. This score will be used to find the highest scoring tree. For more details on how standard RNNs can be used for parsing, see Socher et al. (2011b).

The standard RNN requires a single composition function to capture all types of compositions: adjectives and nouns, verbs and nouns, adverbs and adjectives, etc. Even though this function is a powerful one, we find a single neural network weight matrix cannot fully capture the richness of compositionality. Several extensions are possible: A two-layered RNN would provide more expressive power, however, it is much harder to train because the resulting neural network becomes very deep and suffers from vanishing gradient problems. Socher et al. (2012) proposed to give every single word a matrix and a vector. The matrix is then applied to the sibling node’s vector during the composition. While this results in a powerful composition function that essentially depends on the words being combined, the number of model parameters explodes and the composition functions do not capture the syntactic commonalities between similar POS tags or syntactic categories.

Based on the above considerations, we propose the Compositional Vector Grammar (CVG) that conditions the composition function at each node on discrete syntactic categories extracted from a

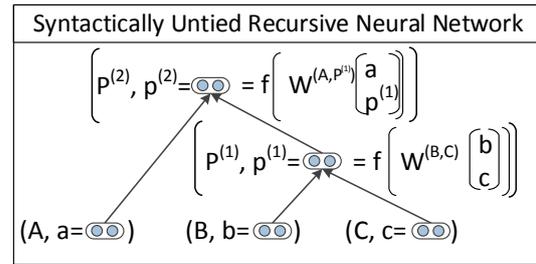


Figure 3: Example of a syntactically untied RNN in which the function to compute a parent vector depends on the syntactic categories of its children which we assume are given for now.

PCFG. Hence, CVGs combine discrete, syntactic rule probabilities and continuous vector compositions. The idea is that the syntactic categories of the children determine what composition function to use for computing the vector of their parents. While not perfect, a dedicated composition function for each rule RHS can well capture common composition processes such as adjective or adverb modification versus noun or clausal complementation. For instance, it could learn that an NP should be similar to its head noun and little influenced by a determiner, whereas in an adjective modification both words considerably determine the meaning of a phrase. The original RNN is parameterized by a single weight matrix  $W$ . In contrast, the CVG uses a syntactically untied RNN (SU-RNN) which has a set of such weights. The size of this set depends on the number of sibling category combinations in the PCFG.

Fig. 3 shows an example SU-RNN that computes parent vectors with syntactically untied weights. The CVG computes the first parent vector via the SU-RNN:

$$p^{(1)} = f \left( W^{(B,C)} \begin{bmatrix} b \\ c \end{bmatrix} \right),$$

where  $W^{(B,C)} \in \mathbb{R}^{n \times 2n}$  is now a matrix that depends on the categories of the two children. In this bottom up procedure, the score for each node consists of summing two elements: First, a single linear unit that scores the parent vector and second, the log probability of the PCFG for the rule that combines these two children:

$$s(p^{(1)}) = (v^{(B,C)})^T p^{(1)} + \log P(P_1 \rightarrow B C), \quad (4)$$

where  $P(P_1 \rightarrow B C)$  comes from the PCFG. This can be interpreted as the log probability of a discrete-continuous rule application with the following factorization:

$$\begin{aligned} P((P_1, p_1) \rightarrow (B, b)(C, c)) \\ = P(p_1 \rightarrow b \mid P_1 \rightarrow B C)P(P_1 \rightarrow B C), \end{aligned} \quad (5)$$

Note, however, that due to the continuous nature of the word vectors, the probability of such a CVG rule application is not comparable to probabilities provided by a PCFG since the latter sum to 1 for all children.

Assuming that node  $p_1$  has syntactic category  $P_1$ , we compute the second parent vector via:

$$p^{(2)} = f \left( W^{(A, P_1)} \begin{bmatrix} a \\ p^{(1)} \end{bmatrix} \right).$$

The score of the last parent in this trigram is computed via:

$$s(p^{(2)}) = (v^{(A, P_1)})^T p^{(2)} + \log P(P_2 \rightarrow A P_1).$$

### 3.4 Parsing with CVGs

The above scores (Eq. 4) are used in the search for the correct tree for a sentence. The goodness of a tree is measured in terms of its score and the CVG score of a complete tree is the sum of the scores at each node:

$$s(\text{CVG}(\theta, x, \hat{y})) = \sum_{d \in N(\hat{y})} s(p^d). \quad (6)$$

The main objective function in Eq. 3 includes a maximization over all possible trees  $\max_{\hat{y} \in Y(x)}$ . Finding the global maximum, however, cannot be done efficiently for longer sentences nor can we use dynamic programming. This is due to the fact that the vectors break the independence assumptions of the base PCFG. A (category, vector) node representation is dependent on all the words in its span and hence to find the true global optimum, we would have to compute the scores for all binary trees. For a sentence of length  $n$ , there are  $\text{Catalan}(n)$  many possible binary trees which is very large even for moderately long sentences.

One could use a bottom-up beam search, keeping a  $k$ -best list at every cell of the chart, possibly for each syntactic category. This beam search inference procedure is still considerably slower than using only the simplified base PCFG, especially since it has a small state space (see next section for

details). Since each probability look-up is cheap but computing SU-RNN scores requires a matrix product, we would like to reduce the number of SU-RNN score computations to only those trees that require semantic information. We note that labeled F1 of the Stanford PCFG parser on the test set is 86.17%. However, if one used an oracle to select the best tree from the top 200 trees that it produces, one could get an F1 of 95.46%.

We use this knowledge to speed up inference via two bottom-up passes through the parsing chart. During the first one, we use only the base PCFG to run CKY dynamic programming through the tree. The  $k = 200$ -best parses at the top cell of the chart are calculated using the efficient algorithm of (Huang and Chiang, 2005). Then, the second pass is a beam search with the full CVG model (including the more expensive matrix multiplications of the SU-RNN). This beam search only considers phrases that appear in the top 200 parses. This is similar to a re-ranking setup but with one main difference: the SU-RNN rule score computation at each node still only has access to its child vectors, not the whole tree or other global features. This allows the second pass to be very fast. We use this setup in our experiments below.

### 3.5 Training SU-RNNs

The full CVG model is trained in two stages. First the base PCFG is trained and its top trees are cached and then used for training the SU-RNN conditioned on the PCFG. The SU-RNN is trained using the objective in Eq. 3 and the scores as exemplified by Eq. 6. For each sentence, we use the method described above to efficiently find an approximation for the optimal tree.

To minimize the objective we want to increase the scores of the correct tree’s constituents and decrease the score of those in the highest scoring incorrect tree. Derivatives are computed via backpropagation through structure (BTS) (Goller and Küchler, 1996). The derivative of tree  $i$  has to be taken with respect to all parameter matrices  $W^{(AB)}$  that appear in it. The main difference between backpropagation in standard RNNs and SU-RNNs is that the derivatives at each node only add to the overall derivative of the specific matrix at that node. For more details on backpropagation through RNNs, see Socher et al. (2010)

### 3.6 Subgradient Methods and AdaGrad

The objective function is not differentiable due to the hinge loss. Therefore, we generalize gradient ascent via the subgradient method (Ratliff et al., 2007) which computes a gradient-like direction. Let  $\theta = (X, W^{(\cdot)}, v^{(\cdot)}) \in \mathbb{R}^M$  be a vector of all  $M$  model parameters, where we denote  $W^{(\cdot)}$  as the set of matrices that appear in the training set. The subgradient of Eq. 3 becomes:

$$\frac{\partial J}{\partial \theta} = \sum_i \frac{\partial s(x_i, \hat{y}_{\max})}{\partial \theta} - \frac{\partial s(x_i, y_i)}{\partial \theta} + \theta,$$

where  $\hat{y}_{\max}$  is the tree with the highest score. To minimize the objective, we use the diagonal variant of AdaGrad (Duchi et al., 2011) with minibatches. For our parameter updates, we first define  $g_\tau \in \mathbb{R}^{M \times 1}$  to be the subgradient at time step  $\tau$  and  $G_t = \sum_{\tau=1}^t g_\tau g_\tau^T$ . The parameter update at time step  $t$  then becomes:

$$\theta_t = \theta_{t-1} - \alpha (\text{diag}(G_t))^{-1/2} g_t, \quad (7)$$

where  $\alpha$  is the learning rate. Since we use the diagonal of  $G_t$ , we only have to store  $M$  values and the update becomes fast to compute: At time step  $t$ , the update for the  $i$ 'th parameter  $\theta_{t,i}$  is:

$$\theta_{t,i} = \theta_{t-1,i} - \frac{\alpha}{\sqrt{\sum_{\tau=1}^t g_{\tau,i}^2}} g_{t,i}. \quad (8)$$

Hence, the learning rate is adapting differently for each parameter and rare parameters get larger updates than frequently occurring parameters. This is helpful in our setting since some  $W$  matrices appear in only a few training trees. This procedure found much better optima (by  $\approx 3\%$  labeled F1 on the dev set), and converged more quickly than L-BFGS which we used previously in RNN training (Socher et al., 2011a). Training time is roughly 4 hours on a single machine.

### 3.7 Initialization of Weight Matrices

In the absence of any knowledge on how to combine two categories, our prior for combining two vectors is to average them instead of performing a completely random projection. Hence, we initialize the binary  $W$  matrices with:

$$W^{(\cdot)} = 0.5[I_{n \times n} I_{n \times n} 0_{n \times 1}] + \epsilon,$$

where we include the bias in the last column and the random variable is uniformly distributed:  $\epsilon \sim$

$\mathcal{U}[-0.001, 0.001]$ . The first block is multiplied by the left child and the second by the right child:

$$\begin{aligned} W^{(AB)} \begin{bmatrix} a \\ b \\ 1 \end{bmatrix} &= \begin{bmatrix} W^{(A)} W^{(B)} \text{bias} \end{bmatrix} \begin{bmatrix} a \\ b \\ 1 \end{bmatrix} \\ &= W^{(A)} a + W^{(B)} b + \text{bias}. \end{aligned}$$

## 4 Experiments

We evaluate the CVG in two ways: First, by a standard parsing evaluation on Penn Treebank WSJ and then by analyzing the model errors in detail.

### 4.1 Cross-validating Hyperparameters

We used the first 20 files of WSJ section 22 to cross-validate several model and optimization choices. The base PCFG uses simplified categories of the Stanford PCFG Parser (Klein and Manning, 2003a). We decreased the state splitting of the PCFG grammar (which helps both by making it less sparse and by reducing the number of parameters in the SU-RNN) by adding the following options to training: ‘noRightRec -dominatesV 0 -baseNP 0’. This reduces the number of states from 15,276 to 12,061 states and 602 POS tags. These include split categories, such as parent annotation categories like  $VP \hat{S}$ . Furthermore, we ignore all category splits for the SU-RNN weights, resulting in 66 unary and 882 binary child pairs. Hence, the SU-RNN has 66+882 transformation matrices and scoring vectors. Note that any PCFG, including latent annotation PCFGs (Matsuzaki et al., 2005) could be used. However, since the vectors will capture lexical and semantic information, even simple base PCFGs can be substantially improved. Since the computational complexity of PCFGs depends on the number of states, a base PCFG with fewer states is much faster.

Testing on the full WSJ section 22 dev set (1700 sentences) takes roughly 470 seconds with the simple base PCFG, 1320 seconds with our new CVG and 1600 seconds with the currently published Stanford factored parser. Hence, increased performance comes also with a speed improvement of approximately 20%.

We fix the same regularization of  $\lambda = 10^{-4}$  for all parameters. The minibatch size was set to 20. We also cross-validated on AdaGrad’s learning rate which was eventually set to  $\alpha = 0.1$  and word vector size. The 25-dimensional vectors provided by Turian et al. (2010) provided the best

Parser	dev (all)	test $\leq$ 40	test (all)
Stanford PCFG	85.8	86.2	85.5
Stanford Factored	87.4	87.2	86.6
Factored PCFGs	89.7	90.1	89.4
Collins			87.7
SSN (Henderson)			89.4
Berkeley Parser			90.1
CVG (RNN)	85.7	85.1	85.0
CVG (SU-RNN)	91.2	91.1	90.4
Charniak-SelfTrain			91.0
Charniak-RS			92.1

Table 1: Comparison of parsers with richer state representations on the WSJ. The last line is the self-trained re-ranked Charniak parser.

performance and were faster than 50-, 100- or 200-dimensional ones. We hypothesize that the larger word vector sizes, while capturing more semantic knowledge, result in too many SU-RNN matrix parameters to train and hence perform worse.

## 4.2 Results on WSJ

The dev set accuracy of the best model is 90.93% labeled F1 on all sentences. This model resulted in 90.44% on the final test set (WSJ section 23). Table 1 compares our results to the two Stanford parser variants (the unlexicalized PCFG (Klein and Manning, 2003a) and the factored parser (Klein and Manning, 2003b)) and other parsers that use richer state representations: the Berkeley parser (Petrov and Klein, 2007), Collins parser (Collins, 1997), SSN: a statistical neural network parser (Henderson, 2004), Factored PCFGs (Hall and Klein, 2012), Charniak-SelfTrain: the self-training approach of McClosky et al. (2006), which bootstraps and parses additional large corpora multiple times, Charniak-RS: the state of the art self-trained and discriminatively re-ranked Charniak-Johnson parser combining (Charniak, 2000; McClosky et al., 2006; Charniak and Johnson, 2005). See Kummerfeld et al. (2012) for more comparisons. We compare also to a standard RNN ‘CVG (RNN)’ and to the proposed CVG with SU-RNNs.

## 4.3 Model Analysis

**Analysis of Error Types.** Table 2 shows a detailed comparison of different errors. We use the code provided by Kummerfeld et al. (2012) and compare to the previous version of the Stanford factored parser as well as to the Berkeley and Charniak-reranked-self-trained parsers (defined above). See Kummerfeld et al. (2012) for details and comparisons to other parsers. One of

Error Type	Stanford	CVG	Berkeley	Char-RS
PP Attach	1.02	0.79	0.82	0.60
Clause Attach	0.64	0.43	0.50	0.38
Diff Label	0.40	0.29	0.29	0.31
Mod Attach	0.37	0.27	0.27	0.25
NP Attach	0.44	0.31	0.27	0.25
Co-ord	0.39	0.32	0.38	0.23
1-Word Span	0.48	0.31	0.28	0.20
Unary	0.35	0.22	0.24	0.14
NP Int	0.28	0.19	0.18	0.14
Other	0.62	0.41	0.41	0.50

Table 2: Detailed comparison of different parsers.

the largest sources of improved performance over the original Stanford factored parser is in the correct placement of PP phrases. When measuring only the F1 of parse nodes that include at least one PP child, the CVG improves the Stanford parser by 6.2% to an F1 of 77.54%. This is a 0.23 reduction in the average number of bracket errors per sentence. The ‘Other’ category includes VP, PRN and other attachments, appositives and internal structures of modifiers and QPs.

**Analysis of Composition Matrices.** An analysis of the norms of the binary matrices reveals that the model learns a soft vectorized notion of head words: Head words are given larger weights and importance when computing the parent vector: For the matrices combining siblings with categories VP:PP, VP:NP and VP:PRT, the weights in the part of the matrix which is multiplied with the VP child vector dominates. Similarly NPs dominate DTs. Fig. 5 shows example matrices. The two strong diagonals are due to the initialization described in Sec. 3.7.

**Semantic Transfer for PP Attachments.** In this small model analysis, we use two pairs of sentences that the original Stanford parser and the CVG did not parse correctly after training on the WSJ. We then continue to train both parsers on two similar sentences and then analyze if the parsers correctly transferred the knowledge. The training sentences are *He eats spaghetti with a fork.* and *She eats spaghetti with pork.* The very similar test sentences are *He eats spaghetti with a spoon.* and *He eats spaghetti with meat.* Initially, both parsers incorrectly attach the PP to the verb in both test sentences. After training, the CVG parses both correctly, while the factored Stanford parser incorrectly attaches both PPs to *spaghetti*. The CVG’s ability to transfer the correct PP attachments is due to the semantic word vector similarity between the words in the sentences. Fig. 4 shows the outputs of the two parsers.

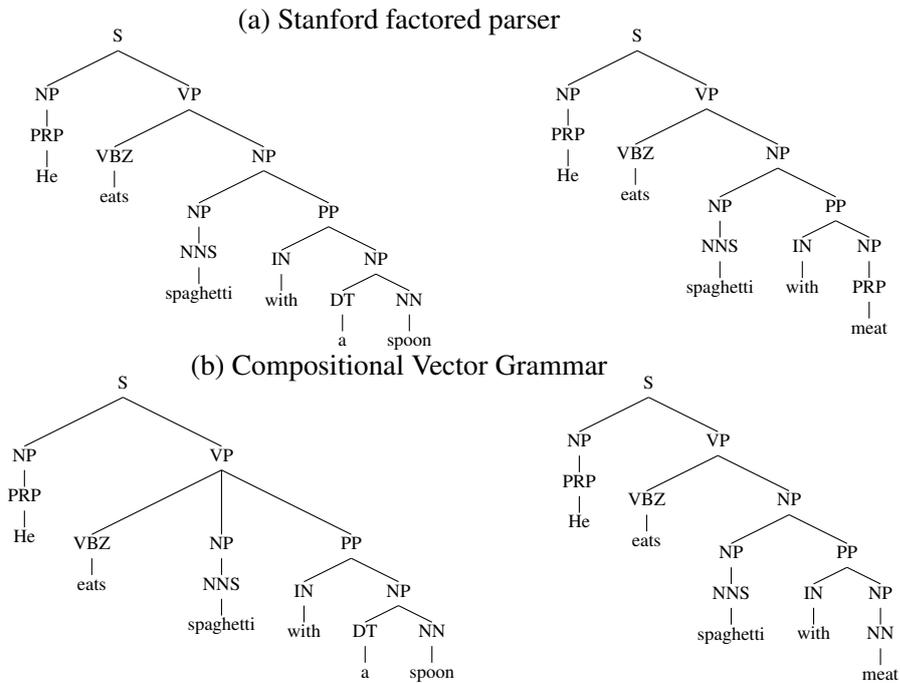


Figure 4: Test sentences of semantic transfer for PP attachments. The CVG was able to transfer semantic word knowledge from two related training sentences. In contrast, the Stanford parser could not distinguish the PP attachments based on the word semantics.

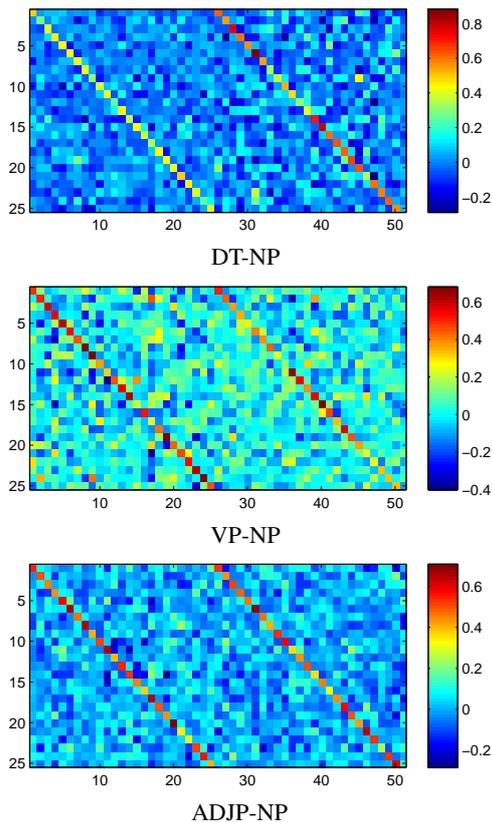


Figure 5: Three binary composition matrices showing that head words dominate the composition. The model learns to not give determiners much importance. The two diagonals show clearly the two blocks that are multiplied with the left and right children, respectively.

## 5 Conclusion

We introduced Compositional Vector Grammars (CVGs), a parsing model that combines the speed of small-state PCFGs with the semantic richness of neural word representations and compositional phrase vectors. The compositional vectors are learned with a new syntactically untied recursive neural network. This model is linguistically more plausible since it chooses different composition functions for a parent node based on the syntactic categories of its children. The CVG obtains 90.44% labeled F1 on the full WSJ test set and is 20% faster than the previous Stanford parser.

## Acknowledgments

We thank Percy Liang for chats about the paper. Richard is supported by a Microsoft Research PhD fellowship. The authors gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-13-2-0040, and the DARPA Deep Learning program under contract number FA8650-10-C-7020. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, AFRL, or the US government.

## References

- Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.
- P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18.
- C. Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of EMNLP*, pages 196–205.
- E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *ACL*.
- E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of ACL*, pages 132–139.
- M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *ACL*.
- M. Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.
- R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167.
- F. Costa, P. Frasconi, V. Lombardo, and G. Soda. 2003. Towards incremental parsing of natural language using recursive neural networks. *Applied Intelligence*.
- J. Duchi, E. Hazan, and Y. Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12, July.
- J. L. Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7(2-3):195–225.
- J. R. Finkel, A. Kleeman, and C. D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of ACL*, pages 959–967.
- D. Gildea and M. Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of ACL*, pages 239–246.
- C. Goller and A. Küchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of the International Conference on Neural Networks*.
- J. Goodman. 1998. *Parsing Inside-Out*. Ph.D. thesis, MIT.
- D. Hall and D. Klein. 2012. Training factored pcfgs with expectation propagation. In *EMNLP*.
- J. Henderson. 2003. Neural network probability estimation for broad coverage parsing. In *Proceedings of EACL*.
- J. Henderson. 2004. Discriminative training of a neural network statistical parser. In *ACL*.
- Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT 2005)*.
- E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. 2012. Improving Word Representations via Global Context and Multiple Word Prototypes. In *ACL*.
- D. Kartsaklis, M. Sadrzadeh, and S. Pulman. 2012. A unified sentence space for categorical distributional-compositional semantics: Theory and experiments. *Proceedings of 24th International Conference on Computational Linguistics (COLING): Posters*.
- D. Klein and C. D. Manning. 2003a. Accurate unlexicalized parsing. In *Proceedings of ACL*, pages 423–430.
- D. Klein and C.D. Manning. 2003b. Fast exact inference with a factored model for natural language parsing. In *NIPS*.
- J. K. Kummerfeld, D. Hall, J. R. Curran, and D. Klein. 2012. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *EMNLP*.
- Q. V. Le, J. Ngiam, Z. Chen, D. Chia, P. W. Koh, and A. Y. Ng. 2010. Tiled convolutional neural networks. In *NIPS*.
- T. Matsuzaki, Y. Miyao, and J. Tsujii. 2005. Probabilistic cfg with latent annotations. In *ACL*.
- D. McClosky, E. Charniak, and M. Johnson. 2006. Effective self-training for parsing. In *NAACL*.
- S. Menchetti, F. Costa, P. Frasconi, and M. Pontil. 2005. Wide coverage natural language processing using kernel methods and neural networks for structured data. *Pattern Recognition Letters*, 26(12):1896–1906.
- T. Mikolov, W. Yih, and G. Zweig. 2013. Linguistic regularities in continuous spaceword representations. In *HLT-NAACL*.
- S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In *NAACL*.
- S. Petrov, L. Barrett, R. Thibaux, and D. Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of ACL*, pages 433–440.
- N. Ratliff, J. A. Bagnell, and M. Zinkevich. 2007. (On-line) subgradient methods for structured prediction. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- R. Socher, C. D. Manning, and A. Y. Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

- R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning. 2011a. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *NIPS*. MIT Press.
- R. Socher, C. Lin, A. Y. Ng, and C.D. Manning. 2011b. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *ICML*.
- R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *EMNLP*.
- B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. 2004. Max-margin parsing. In *Proceedings of EMNLP*, pages 1–8.
- I. Titov and J. Henderson. 2006. Porting statistical parsers with data-defined kernels. In *CoNLL-X*.
- I. Titov and J. Henderson. 2007. Constituent parsing with incremental sigmoid belief networks. In *ACL*.
- J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, pages 384–394.
- P. D. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

# Discriminative state tracking for spoken dialog systems

Angeliki Metallinou<sup>1\*</sup>, Dan Bohus<sup>2</sup>, and Jason D. Williams<sup>2</sup>

<sup>1</sup>University of Southern California, Los Angeles, CA, USA

<sup>2</sup>Microsoft Research, Redmond, WA, USA

metallin@usc.edu dbohush@microsoft.com jason.williams@microsoft.com

## Abstract

In spoken dialog systems, statistical *state tracking* aims to improve robustness to speech recognition errors by tracking a posterior distribution over hidden dialog states. Current approaches based on generative or discriminative models have different but important shortcomings that limit their accuracy. In this paper we discuss these limitations and introduce a new approach for discriminative state tracking that overcomes them by leveraging the problem structure. An offline evaluation with dialog data collected from real users shows improvements in both state tracking accuracy and the quality of the posterior probabilities. Features that encode speech recognition error patterns are particularly helpful, and training requires relatively few dialogs.

## 1 Introduction

Spoken dialog systems interact with users via natural language to help them achieve a goal. As the interaction progresses, the dialog manager maintains a representation of the state of the dialog in a process called *dialog state tracking*. For example, in a bus schedule information system, the dialog state might indicate the user's desired bus route, origin, and destination. Dialog state tracking is difficult because automatic speech recognition (ASR) and spoken language understanding (SLU) errors are common, and can cause the system to misunderstand the user's needs. At the same time, state tracking is crucial because the system relies on the estimated dialog state to choose actions – for example, which bus schedule

information to present to the user.

The dialog state tracking problem can be formalized as follows (Figure 1). Each system turn in the dialog is one datapoint. For each datapoint, the input consists of three items: a set of  $K$  features that describes the current dialog context,  $G$  dialog state hypotheses, and for each dialog state hypothesis,  $M$  features that describe that dialog state hypothesis. The task is to assign a probability distribution over the  $G$  dialog state hypotheses, plus a meta-hypothesis which indicates that none of the  $G$  hypotheses is correct.

Note that  $G$  varies across turns (datapoints) – for example, in the first turn of Figure 1,  $G = 3$ , and in the second and third turns  $G = 5$ . Also note that the dialog state tracker is *not* predicting the contents of the dialog state hypotheses; the dialog state hypotheses contents are given by some external process, and the task is to predict a probability distribution over them, where the probability assigned to a hypothesis indicates the probability that it is correct. It is a requirement that the  $G$  hypotheses are disjoint; with the special “everything else” meta-hypothesis, exactly one hypothesis is correct by construction. After the dialog state tracker has output its distribution, this distribution is passed to a separate, downstream process that chooses what action to take next (e.g., how to respond to the user).

Dialog state tracking can be seen an analogous to assigning a probability distribution over items on an ASR N-best list given speech input and the recognition output, including the contents of the N-best list. In this task, the general features describe the recognition overall (such as length of utterance), and the hypothesis-specific features describe each N-best entry (such as decoder cost).

\* Work done while at Microsoft Research

Another analogous task is assigning a probability distribution over a set of URLs given a search query and the URLs. Here, general features describe the whole set of results, e.g., number of words in the query, and hypothesis-specific features describe each URL, e.g., the fraction of query words contained in page.

For dialog state tracking, most commercial systems use hand-crafted heuristics, selecting the SLU result with the highest confidence score, and discarding alternatives. In contrast, statistical approaches compute a posterior *distribution* over many *hypotheses* for the dialog state. The key insight is that dialog is a temporal process in which correlations between turns can be harnessed to overcome SLU errors. Statistical state tracking has been shown to improve task completion in end-to-end spoken dialog systems (Bohus and Rudnicky (2006); Young et al. (2010); Thomson and Young (2010)).

Two types of statistical state tracking approaches have been proposed. *Generative* approaches (Horvitz and Paek (1999); Williams and Young (2007); Young et al. (2010); Thomson and Young (2010)) use generative models that capture how the SLU results are generated from hidden dialog states. These models can be used to track an arbitrary number of state hypotheses, but cannot easily incorporate large sets of potentially informative features (e.g. from ASR, SLU, dialog history), resulting in poor probability estimates. As an illustration, in Figure 1, a generative model might fail to assign the highest score to the correct hypothesis (61C) after the second turn. In contrast, *discriminative* approaches use conditional models, trained in a discriminative fashion (Bohus and Rudnicky (2006)) to directly estimate the distribution over a set of state hypotheses based on a large set of informative features. They generally produce more accurate distributions, but in their current form they can only track a handful of state hypotheses. As a result, the correct hypothesis may be discarded: for instance, in Figure 1, a discriminative model might consider only the top 2 SLU results, and thus fail to consider the correct 61C hypothesis at all.

The main contribution of this paper is to develop a new discriminative model for dialog state tracking that can operate over an arbitrary number of hypotheses *and* still compute accurate probability estimates. We also explore the relative im-

portance of different feature sets for this task, and measure the amount of data required to reliably train our model.

## 2 Data and experimental design

We use data from the public deployment of two systems in the Spoken Dialog Challenge (Black et al. (2010)) which provide bus schedule information for Pittsburgh, USA. The systems, DS1 and DS2, were fielded by AT&T, and are described in Williams et al. (2010) and Williams (2012). Both systems followed a highly directed flow, separately collecting 5 *slots*. All users were asked for their bus route, origin, and destination; then, they were optionally prompted for a date and time. Each slot was explicitly or implicitly confirmed before collecting the next. At the end, bus times were presented. The two systems differed in acoustic models, confidence scoring model, state tracking method and parameters, number of supported routes (8 vs 40, for DS1 and DS2 respectively), presence of minor bugs, and user population. These differences yield distinctions in the distributions in the two corpora (Williams (2012)).

In both systems, a dialog state hypothesis consists of a value of the user’s goal for a certain slot: for example, a state hypothesis for the origin slot might be “carnegie mellon university”. The number  $G$  of state hypotheses (e.g. slot values) observed so far depends on the dialog, and turn within that dialog. For instance, in Fig. 1,  $G$  progressively takes values 3, 5 and 5. Dialog state hypotheses with identical contents (e.g., the same bus route) are merged. The correctness of the SLU results was manually labeled by professional annotators.

### 2.1 Experimental setup

To perform a comparative analysis of various state tracking algorithms, we test them *offline*, i.e., by re-running state tracking against the SLU results from deployment. However, care must be taken: when the improved state-tracker is installed into a dialog system and used to drive action selection, the distribution of the resulting dialog data (which is an input for the state tracker) will change. In other words, it is known *a priori* that the train and test distributions will be *mismatched*. Hence, when conducting offline experiments, if train and test data were drawn from the same *matched* distribution, this may overstate performance.

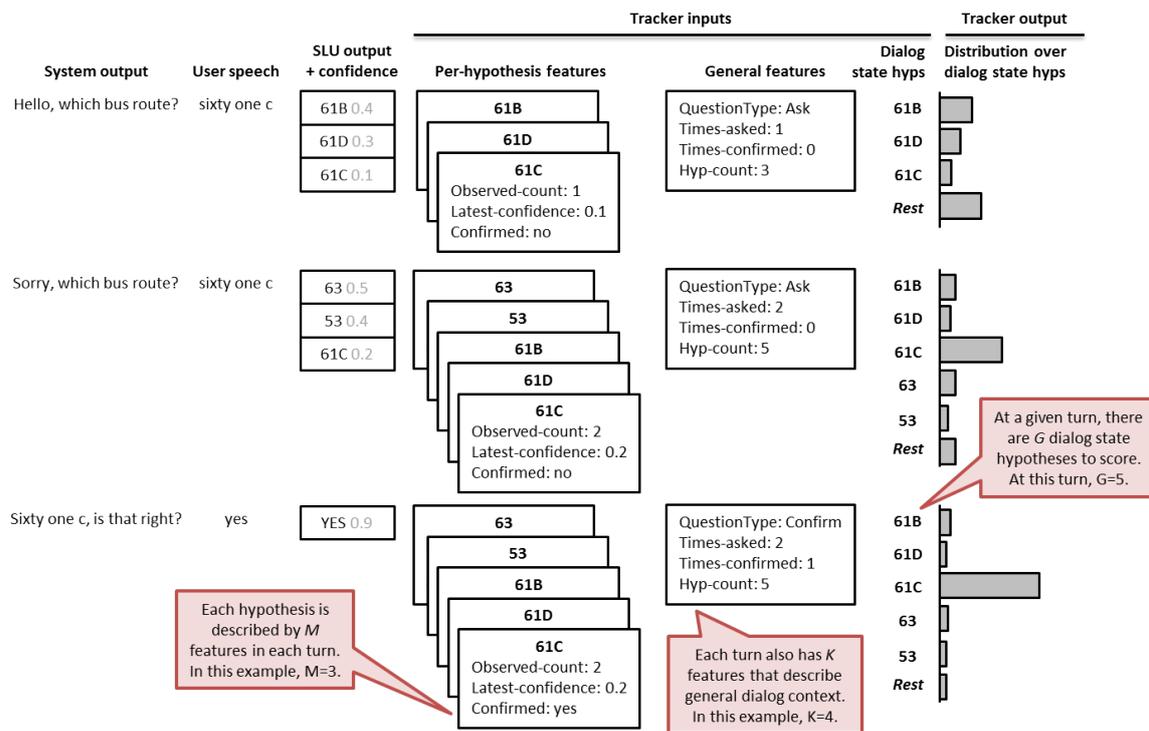


Figure 1: Overview of dialog state tracking. In this example, the dialog state contains the user’s desired bus route. At each turn, the system produces a spoken output. The user’s spoken response is processed to extract a set of spoken language understanding (SLU) results, each with a *local* confidence score. A set of  $G$  dialog state hypotheses is formed by considering all SLU results observed so far, including the current turn and all previous turns. For each state hypothesis, a feature extractor produces a set of  $M$  hypothesis-specific features, plus a single set of  $K$  general features that describes the current dialog context. The dialog state tracker uses these features to produce a distribution over the  $G$  state hypotheses, plus a meta-hypothesis *rest* which accounts for the possibility that none of the  $G$  hypotheses are correct.

dataset	train set	test set
MATCH1	half calls from DS2	remaining calls in DS2
MATCH2	half calls from DS1, half from DS2	remaining calls from DS1 and DS2
MISMATCH	all calls from DS1	all calls from DS2

Table 1: Train-test data splits

To account for this effect, we explicitly study train/test mismatch through three partitions of data from DS1 and DS2 (see Table 1): MATCH1 contains matched train/test data from the DS2 dataset; MATCH2 contains matched train/test data from both datasets; finally, MISMATCH contains mismatched train/test data. While the MISMATCH condition may not identically replicate the mismatch observed from deploying a new state tracker *online* (since online characteristics depend on user behavior) training on DS1 and testing on DS2 at least ensures the presence of some real-world mismatch.

We assess performance via two metrics: *accuracy* and *L2 norm*. Accuracy indicates whether the state hypothesis with the highest assigned probability is correct, where *rest* is correct iff none of the SLU results prior to the current turn include the user’s goal. High accuracy is important as a dialog system must ultimately commit to a single interpretation of the user’s needs – e.g., it must commit to a route in order to provide bus timetable information. In addition, the L2 norm (or Brier score, Murphy (1973)) also captures how well calibrated the output probabilities are, which is crucial to decision theoretic methods for action selection. The L2 norm is computed between the output posterior and the ground-truth vector, which has 1 in the position of the correct item and 0 elsewhere. Both metrics are computed for each slot in each turn, and reported by averaging across all turns and slots.

## 2.2 Hand-crafted baseline state tracker

As a baseline, we construct a hand-crafted state tracking rule that follows a strategy common in commercial systems: it returns the SLU result with the maximum confidence score, ignoring all other hypotheses. Although this is very a simple rule, it is very often effective. For example, if the user says “no” to an explicit confirmation or “go back” to an implicit confirmation, they are asked the same question again, which gives an opportunity for a higher confidence score. Of the  $G$  possible hypotheses for a slot, we denote the number actually assigned a score by a model as  $\tilde{G}$ , so in this heuristic baseline  $\tilde{G} = 1$ .

The performance of this baseline (BASELINE in Table 3) is relatively strong because the top SLU result is by far most likely to be correct, and because the confidence score was already trained with slot-specific speech data (Williams and Balakrishnan (2009), Williams (2012)). However, this simple rule can’t make use of SLU results on the N-best list, or statistical priors; these limitations motivate the use of statistical state trackers, introduced next.

## 3 Generative state tracking

Generative state tracking approaches leverage models that describe how SLU results are *generated* from a hidden dialog state, denoted  $g$ . The user’s true (unobserved) action  $u$  is conditioned on  $g$  and the system action  $a$  via a user action model  $P(u|g, a)$ , and also on the observed SLU result  $\tilde{u}$  via a model of how SLU results are generated  $P(\tilde{u}|u)$ . Given a prior distribution  $b(g)$  and a result  $\tilde{u}$ , an updated distribution  $b'(g)$  can be computed by summing over all hidden user actions  $u$ :

$$b'(g) = \eta \sum_u P(\tilde{u}|u) \cdot P(u|g, a) b(g) \quad (1)$$

where  $\eta$  is a normalizing constant (Williams and Young (2007)). Generative approaches model the posterior over *all* possible dialog state hypotheses, including those not observed in the SLU N-best lists. In general this is computationally intractable because the number of states is too large. One approach to scaling up is to group  $g$  into a few *partitions*, and to track only states suggested by observed SLU results (Young et al. (2010); Williams (2010); Gašić and Young (2011)). Another approach is to *factor* the components of a dialog

state, make assumptions about conditional independence between the components, and apply approximate inference techniques such as loopy belief propagation (Thomson and Young (2010)).

In deployment, DS1 and DS2 used the AT&T Statistical Dialog Toolkit (ASDT) for dialog state tracking (Williams (2010); AT&T Statistical Dialog Toolkit). ASDT implements a generative update of the form of Eq 1, and uses partitions to maintain tractability. Component models were learned from dialog data from a different dialog system. A maximum of  $\tilde{G} = 20$  state hypotheses were tracked for each slot. The performance (GENONLINE in Table 3), was worse than BASELINE: an in-depth analysis attributed this to the mismatch between train and test data in the component models, and to the underlying flawed assumption of eq. 1 that observations at different turns are independent conditioned on the dialog state – in practice, confusions made by speech recognition are highly correlated (Williams (2012)).

For all datasets, we re-estimated the models on the train set and re-ran generative tracking with an unlimited number of partitions (i.e.,  $\tilde{G} = G$ ); see GENOFFLINE in Table 3. The re-estimated tracker improved accuracy in MATCH conditions, but degraded accuracy in the MISMATCH condition. This can be partly attributed to the difficulty in estimating accurate initial priors  $b(g)$  for MISMATCH, where the bus route, origin, and destination slot values in train and test systems differed significantly.

## 4 Discriminative State Tracking: Preliminaries and existing work

In contrast to generative models, discriminative approaches to dialog state tracking directly predict the correct state hypothesis by leveraging discriminatively trained conditional models of the form  $b(g) = P(g|f)$ , where  $f$  are features extracted from various sources, e.g. ASR, SLU, dialog history, etc. In this work we will use maximum entropy models. We begin by briefly introducing these models in the next subsection. We then describe the features used, and finally review existing discriminative approaches for state tracking which serve as a starting point for the new approach we introduce in Section 5.

## 4.1 Maximum entropy models

The maximum entropy framework (Berger et al. (1996)) models the conditional probability distribution of the label  $y$  given features  $\mathbf{x}$ ,  $p(y|\mathbf{x})$  via an exponential model of the form:

$$P(y|\mathbf{x}, \lambda) = \frac{\exp(\sum_{i \in I} \lambda_i \phi_i(\mathbf{x}, y))}{\sum_{y \in Y} \exp(\sum_{i \in I} \lambda_i \phi_i(\mathbf{x}, y))} \quad (2)$$

where  $\phi_i(\mathbf{x}, y)$  are feature functions jointly defined on features and labels, and  $\lambda_i$  are the model parameters. The training procedure optimizes the parameters  $\lambda_i$  to maximize the likelihood over the data instances subject to regularization penalties. In this work, we optimize the L1 penalty using a cross-validation process on the train set, and we use a fixed L2 penalty based on heuristic based on the dataset size. The same optimization is used for all models.

## 4.2 Features

Discriminative approaches for state tracking rely on informative features to predict the correct dialog state. In this work we designed a set of *hypothesis-specific* features that convey information about the correctness of a particular state hypothesis, and a set of *general* features that convey information about the correctness of the *rest* meta-hypothesis.

*Hypothesis-specific* features can be grouped into 3 categories: *base*, *history* and *confusion* features. *Base* features consider information about the *current* turn, including rank of the current SLU result (current hypothesis), the SLU result confidence score(s) in the current N-best list, the difference between the current hypothesis score and the best hypothesis score in the current N-best list, etc. *History* features contain additional useful information about past turns. Those include the number of times an SLU result has been observed before, the number of times an SLU result has been observed before at a specific rank such as rank 1, the sum and average of confidence scores of SLU results across all past recognitions, the number of possible past user negations or confirmations of the current SLU result etc.

*Confusion* features provide information about likely ASR errors and confusability. Some recognition results are more likely to be incorrect than others – background noise tends to trigger certain results, especially short bus routes like “p”. Moreover, similar sounding phrases are more likely to

be confused. The confusion features were computed on a subset of the training data. For each SLU result we computed the fraction of the time that the result was correct, and the binomial 95% confidence interval for that estimate. Those two statistics were pre-computed for all SLU results in the training data subset, and were stored in a lookup table. At runtime, when an SLU hypothesis is recognized, its statistics from this lookup table are used as features. Similar statistics were computed for prior probability of an SLU result appearing on an N-best list, and prior probability of SLU result appearance at specific rank positions of an N-best list, prior probability of confusion between pairs of SLU results, and others.

*General* features provide aggregate information about dialog history and SLU results, and are shared across different SLU results of an N-best list. For example, from the current turn, we use the number of distinct SLU results, the entropy of the confidence scores, the best path score of the word confusion network, etc. We also include features that contain aggregate information about the sequence of all N-best lists up to the current turn, such as the mean and variance of N-best list lengths, the number of distinct SLU results observed so far, the entropy of their corresponding confidence scores, and others.

We denote the number of *hypothesis-specific* features as  $M$ , and the number of *general* features as  $K$ .  $K$  and  $M$  are each in the range of 100–200, although  $M$  varies depending on whether *history* and *confusion* features are included. For a given dialog turn with  $G$  state hypotheses, there are a total of  $G * M + K$  distinct features.

## 4.3 Fixed-length discriminative state tracking

In past work, Bohus and Rudnicky (2006) introduced discriminative state tracking, casting the problem as standard multiclass classification. In this setup, each turn constitutes one data instance. Since in dialog state tracking the number of state hypotheses varies across turns, Bohus and Rudnicky (2006) chose a *subset* of  $\tilde{G}$  state hypotheses to score. In this work we used a similar setup, where we considered the top  $G_1$  SLU results from the current N-best list at turn  $t$ , and the top  $G_2$  and  $G_3$  SLU results from the previous N-best lists at turns  $t - 1$  and  $t - 2$ . The problem can then be formulated as multiclass classification

over  $\tilde{G} + 1 = G_1 + G_2 + G_3 + 1$  classes, where the correct class indicates which of these hypotheses (or *rest*) is correct. We experimented with different values and found that  $G_1 = 3$ ,  $G_2 = 2$ , and  $G_3 = 1$  ( $\tilde{G} = 6$ ) yielded the best performance.

Feature functions are defined in the standard way, with one feature function  $\phi$  and weight  $\lambda$  for each (feature,class) pair. Formally,  $\phi$  of eq. 2 is defined as  $\phi_{i,j}(x, y) = x_i \delta(y, j)$ , where  $\delta(y, j) = 1$  if  $y = j$  and 0 otherwise.  $i$  indexes over the  $\tilde{G}M + K$  features and  $j$  over the  $\tilde{G} + 1$  classes.<sup>1</sup> The two-dimensional subscript  $i, j$  if used for clarity of notation, but is otherwise identical in role to the one-dimension subscript  $i$  in Eq 2. Figure 2 illustrates the relationship between hypotheses and weights.

Results are reported as DISCFIXED in Table 3. In the MATCH conditions, performance is generally higher than the other baselines, particularly when confusion features are included. In the MIS-MATCH condition, performance is worse than the BASELINE.

A strength of this approach is that it enables features from every hypothesis to independently affect every class. However, the total number of feature functions (hence weights to learn) is  $(\tilde{G} + 1) \times (\tilde{G}M + K)$ , which increases quadratically with the number of hypotheses considered  $\tilde{G}$ . Although regularization can help avoid overfitting *per se*, it becomes a more challenging task with more features. Learning weights for each (feature,class) pair has the drawback that the effect of hypothesis-specific features such as confidence have to be learned separately for every hypothesis. Also, although we know in advance that posteriors for a dialog state hypothesis are most dependent on the features corresponding to that hypothesis, in this approach the features from *all* hypotheses are pooled together and the model is left to discover these correspondences via learning. Furthermore, items lower down on the SLU N-best list are much less likely to be correct: an item at a very deep position (say 19) might never be correct in the training data – when this occurs, it is unreasonable to expect posteriors to be estimated accurately.

As a result of these issues, in practice  $\tilde{G}$  is limited to being a small number – here we found that increasing  $\tilde{G} > 6$  degraded performance. Yet with

<sup>1</sup>Although in practice, maximum entropy model constraints render weights for one class redundant.

$\tilde{G} = 6$ , we found that in 10% of turns, the correct state hypothesis was present but was being discarded by the model, which substantially reduces the upper-bound on tracker performance. In the next section, we introduce a novel discriminative state tracking approach that addresses the above limitations, and enables jointly considering an arbitrary number of state hypotheses, by exploiting the structure inherent in the dialog state tracking problem.

## 5 Dynamic discriminative state tracking

The key idea in the proposed approach is to use feature functions that *link* hypothesis-specific features to their corresponding dialog state hypothesis. This approach makes it straightforward to model relationships such as “higher confidence for an SLU result increases the probability of its corresponding state hypothesis being correct”. This formulation also decouples the number of model parameters (i.e. weights to learn) from the number of hypotheses considered, allowing an arbitrary number of dialog states hypotheses to be scored.

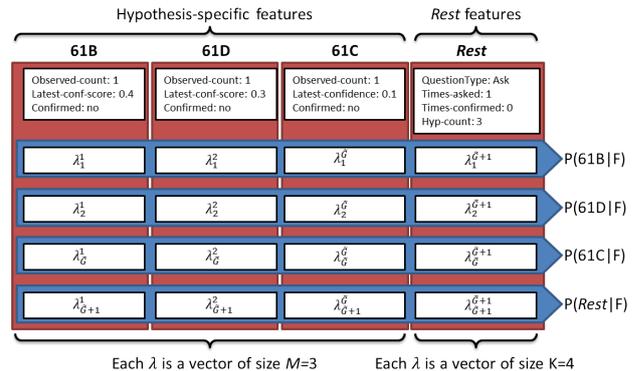


Figure 2: The DISCFIXED model is a traditional maximum entropy model for classification. Every feature in every hypothesis is linked to every hypothesis, requiring  $(\tilde{G} + 1)(\tilde{G}M + K)$  weights.

We begin by re-stating how features are indexed. Recall each dialog state hypothesis has  $M$  hypothesis-specific features; for each hypothesis, we concatenate these  $M$  features with the  $K$  general features, which are identical for all hypotheses. For the meta-hypothesis *rest*, we again concatenate  $M+K$  features, where the  $M$  hypothesis-specific features take special undefined values. We write  $x_i^g$  to refer to the  $i$ th feature of hypothesis  $g$ , where  $i$  ranges from 1 to  $M + K$  and  $g$  from 1 to  $G + 1$ .

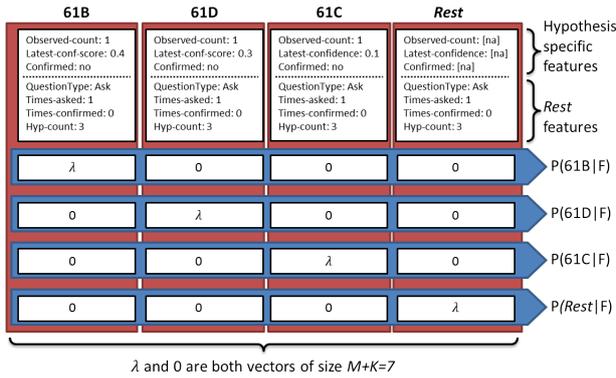


Figure 3: The DISCDYN model presented in this paper exploits the structure of the state tracking problem. Features are linked to only their own hypothesis, and weights are shared across all hypotheses, requiring  $M + K$  weights.

algorithm	description
BASELINE	simple hand-crafted rule
GENONLINE	generative update, in deployed system
GENOFFLINE	generative update, re-trained and run offline
DISCFIXED	discr. fixed size multiclass (7 classes)
DISCDYN1	discr. joint dynamic estimation
DISCDYN2	discr. joint dynamic estimation, using indicator encoding of ordinal features
DISCDYN3	discr. joint dynamic estimation, using indicator encoding and ordinal-ordinal conjunctions
DISCIND	discr. separate estimation

Table 2: Description of the various implemented state tracking algorithms

The model is based on  $M + K$  feature functions. However, unlike in traditional maximum entropy models such as the fixed-position model above, these features functions are *dynamically defined* when presented with each turn. Specifically, for a turn with  $G$  hypotheses, we define  $\phi_i(\mathbf{x}, y = g) = x_i^g$ , where  $y$  ranges over the set of possible dialog states  $G + 1$  (and as above  $i \in 1 \dots M + K$ ). The feature function  $\phi_i$  is dynamic in that the domain of  $y$  – i.e., the number of dialog state hypotheses to score – varies from turn to turn. With feature functions defined this way, standard maximum entropy optimization is then applied to learn the corresponding set of  $M + K$  weights, denoted  $\lambda_i$ . Fig. 3 shows the relationship of hypotheses and weights.

In practice, this formulation – in which *general* features are duplicated across every dialog state hypothesis – may require some additional feature engineering: for every hypothesis  $g$  and *general* feature  $i$ , the value of that general feature  $x_i^g$  will

be multiplied by the same weight  $\lambda_i$ . The result is that any setting of  $\lambda_i$  affects all scores identically, with no net change to the resulting posterior. Nonetheless, *general* features do contain useful information for state tracking; to make use of them, we add conjunctions (combinations) of *general* and *hypothesis-specific* features.

We use 3 different feature variants. In DISCDYN1, we use the original feature set, ignoring the problem described above (so that the *general* features contribute no information), resulting in  $M + K$  weights. DISCDYN2 adds indicator encodings of the ordinal-valued *hypothesis-specific* features. For example, rank is encoded as a vector of boolean indicators, where the first indicator is nonzero if  $rank = 1$ , the second is nonzero if  $rank = 2$ , and the third if  $rank \geq 3$ . This provides a more detailed encoding of the ordinal-valued *hypothesis-specific* features, although it still ignores information from the *general* features. This encoding increases the number of weights to learn to about  $2(M + K)$ .

Finally, DISCDYN3 extends DISCDYN2 by including conjunctions of the ordinal-valued *general* features with ordinal-valued *hypothesis-specific* features. For example, if the 3-way *hypothesis-specific* indicator feature for rank described above were conjoined with a 4-way *general* indicator feature for dialog state, the result would be an indicator of dimension  $3 \times 4 = 12$ . This expansion results in approximately  $10(M + K)$  weights to learn in DISCDYN3.<sup>2</sup>

For comparison, we also estimated a simpler alternative model, called DISCIND. This model consists of 2 binary classifiers: the first one scores each hypothesis in isolation, using the  $M$  *hypothesis-specific* features for that hypothesis + the  $K$  *general* features for that turn, and outputs a (single) probability that the hypothesis is correct. For this classifier, each hypothesis (not each turn) defines a data instance. The second binary classifier takes the  $K$  *general* features, and outputs a probability that the *rest* meta-hypothesis is correct. For this second classifier, each turn defines one data instance. The output of these two models is then calibrated with isotonic regression (Zadrozny and Elkan (2002)) and normalized to generate the posterior over all hypotheses.

<sup>2</sup>We explored adding all possible conjunctions, including real-valued features, but this increased memory and computational requirements dramatically without performance gains.

Metric Dataset Features	Accuracy (larger numbers better)									L2 (smaller numbers better)								
	MATCH1			MATCH2			MISMATCH			MATCH1			MATCH2			MISMATCH		
	<i>b</i>	<i>bc</i>	<i>bch</i>	<i>b</i>	<i>bc</i>	<i>bch</i>	<i>b</i>	<i>bc</i>	<i>bch</i>	<i>b</i>	<i>bc</i>	<i>bch</i>	<i>b</i>	<i>bc</i>	<i>bch</i>	<i>b</i>	<i>bc</i>	<i>bch</i>
BASELINE	61.5	61.5	61.5	63.4	63.4	63.4	<b>62.5</b>	62.5	62.5	27.1	27.1	27.1	25.5	25.5	25.5	27.3	27.3	27.3
GENONLINE	54.4	54.4	54.4	55.8	55.8	55.8	54.8	54.8	54.8	34.8	34.8	34.8	32.0	32.0	32.0	34.8	34.8	34.8
GENOFFLINE	57.1	57.1	57.1	60.1	60.1	60.1	51.8	51.8	51.8	37.6	37.6	37.6	33.4	33.4	33.4	42.0	42.0	42.0
DISCFIXED	61.9	66.7	65.3	63.6	69.7	68.8	59.1	61.9	59.3	27.2	23.6	24.4	25.8	21.9	22.4	28.9	27.8	27.8
DISCDYN1	62.0	70.9	71.1	64.4	<b>72.4</b>	<b>72.9</b>	59.4	61.8	62.3	26.3	<b>21.3</b>	<b>20.9</b>	25.0	<b>20.4</b>	<b>20.1</b>	27.7	26.3	25.9
DISCDYN2	62.6	<b>71.3</b>	<b>71.5</b>	65.7	72.1	72.2	61.9	63.2	<b>63.1</b>	26.3	21.4	21.2	24.4	20.5	20.4	<b>26.9</b>	<b>25.8</b>	<b>25.4</b>
DISCDYN3	<b>63.6</b>	70.1	70.9	<b>65.9</b>	72.1	70.7	60.7	62.1	62.9	<b>26.2</b>	21.5	21.4	<b>24.3</b>	20.6	20.7	27.1	25.9	26.1
DISCIND	62.4	69.8	70.5	63.4	71.5	71.8	59.9	<b>63.3</b>	62.2	26.7	23.3	22.5	25.7	21.8	20.7	28.4	27.3	28.8

Table 3: Performance of the different algorithms on each dataset using three feature combinations. *Base* features are denoted as *b*, ASR/SLU *confusion* features as *c* and *history* features as *h*. Performance for the feature combinations *bh* is omitted for space; it is between *b* and *bc*.

## 6 Results and discussion

The implemented state tracking methods are summarized in Table 2, and our results are presented in Table 3. These results suggest several conclusions. First, discriminative approaches for state tracking broadly outperform generative methods. Since discriminative methods incorporate many features and are trained directly to optimize performance, this is perhaps unsurprising for the MATCH conditions. It is interesting that discriminative methods are also superior in the more realistic MISMATCH setting, albeit with smaller gains. This result suggests that discriminative methods have good promise when deployed into real systems, where mismatch between training and test distributions is expected.

Second, the dynamic discriminative DISCDYN models also outperformed the fixed-length discriminative methods. This shows the benefit of a model which can score every dialog state hypotheses, rather than a fixed subset. Third, the three variants of the DISCDYN model, which progressively contain more detailed feature encoding and conjunctions, perform similarly. This suggests that a relatively simple encoding is sufficient to achieve good performance, as the feature indicators and conjunctions present in DISCDYN2 and DISCDYN3 give only a small additional increase.

Among the discriminative models, the jointly-optimized DISCDYN versions also slightly outperform the simpler, independently-optimized DISCIND version. This is to be expected, for two reasons: first, DISCIND is trained on a per-hypothesis basis, while the DISCDYN models are trained on a *per-turn* basis, which is the true performance metric. For example, some turns have 1 hypothesis and others have 100, but DISCIND training counts

all hypotheses equally. Second, model parameters in DISCIND are trained independently of competing hypotheses. However, they should rather be adjusted specifically so that the correct item receives a larger score than incorrect items – not merely to increase scores for correct items and decrease scores for incorrect items in isolation – and this is what is done in the DISCDYN models.

The analysis of various feature sets indicates that the ASR/SLU error correlation (*confusion*) features yield the largest improvement – c.f. feature set *bc* compared to *b* in Table 3. The improvement is smallest for MISMATCH, which underscores the challenges of mismatched train and test conditions during a realistic runtime scenario. Note, however, that we have constructed a highly mismatched case where we train on DS1 (that supports just 8 routes) and test on DS2 (that supports 40 routes). Therefore, many route, origin and destination slot values in the test data do not appear in the training data. Hence, it is unsurprising that the positive effect of confusion features would decrease.

While Table 3 shows performance measures averaged across all turns, Table 4 breaks down performance measures *by slot*, using the full feature set *bch* and the realistic MISMATCH dataset. Results here show a large variation in performance across the different slots. For the date and time slots, there is an order of magnitude less data than for the other slots; however performance for dates is quite good, whereas times is rather poor. We believe this is because the SLU confusion features can be estimated well for slots with small cardinalities (there are 7 possible values for the day), and less well for slots with large cardinalities (there are  $24 \times 60 = 1440$  possible time values). This sug-

Accuracy (larger numbers better)					
algorithms	rout	origin	dest.	date	time
BASELINE	53.81	66.49	67.78	71.88	<b>52.32</b>
GENONLINE	50.02	54.11	59.05	75.78	35.02
GENOFFLINE	48.12	58.82	58.98	72.66	20.25
DISCFIXED	52.83	67.81	70.67	71.88	33.34
DISCDYN1	54.28	68.24	68.53	79.69	40.51
DISCDYN2	<b>56.18</b>	68.42	70.10	80.47	40.51
DISCDYN3	54.52	66.24	67.96	<b>82.81</b>	43.04
DISCIND	54.25	<b>68.84</b>	<b>70.79</b>	78.13	38.82

L2 metric (smaller numbers better)					
algorithms	route	origin	dest.	date	time
BASELINE	33.15	24.67	24.68	21.61	<b>32.35</b>
GENONLINE	35.50	35.10	31.13	19.86	52.58
GENOFFLINE	46.42	35.73	37.76	19.97	70.30
DISCFIXED	34.09	23.92	23.35	17.59	40.15
DISCDYN1	31.30	23.01	23.07	15.29	37.02
DISCDYN2	<b>30.53</b>	<b>22.40</b>	<b>22.74</b>	<b>13.58</b>	37.59
DISCDYN3	31.58	23.86	23.68	13.93	37.52
DISCIND	36.50	23.45	23.41	15.20	45.43

Table 4: Performance per slot on dataset MIS-MATCH using the full feature set *bch*.

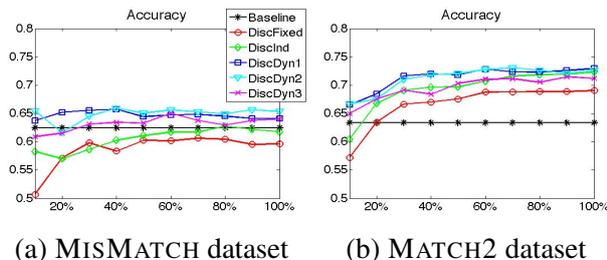


Figure 4: Accuracy vs. amount of training data

gests that the amount of data required to estimate a good model may depend on the cardinality of slot values.

Finally, in Figure 4 we show how performance varies with different amounts of training data for the MATCH2 and MISMATCH datasets, where the full training set size is approximately 5600 and 4400 turns, respectively. In both cases asymptotic performance is reached after about 2000 turns, or about 150 dialogs. This is particularly encouraging, as it suggests models could be learned or adapted online with relatively little data, or could even be individually tailored to particular users.

## 7 Conclusion and Future Work

Dialog state tracking is crucial to the successful operation of spoken dialog systems. Recently developed statistical approaches are promising as they fully utilize the dialog history, and can incorporate priors from past usage data. However,

existing methodologies are either limited in their accuracy or their coverage, both of which hamper performance.

In this paper, we have introduced a new model for discriminative state tracking. The key idea is to exploit the structure of the problem, in which each dialog state hypothesis has features drawn from the same set. In contrast to past approaches to discriminative state tracking which required a number of parameters quadratic in the number of state hypotheses, our approach uses a constant number of parameters, invariant to the number of state hypotheses. This is a crucial property that enables generalization and dealing with an unlimited number of hypotheses, overcoming a key limitation in previous models.

We evaluated the proposed method and compared it to existing generative and discriminative approaches on a corpus of real-world human-computer dialogs chosen to include a mismatch between training and test, as this will be found in deployments. Results show that the proposed model exceeds both the accuracy and probability quality of all baselines when using the richest feature set, which includes information about common ASR confusions and dialog history. The model can be trained efficiently, i.e. only about 150 training dialogs are necessary.

The next step is to incorporate this approach into a deployed dialog system, and use the estimated posterior over dialog states as input to the action selection process. In future, we also hope to explore unsupervised online adaptation, where the trained model can be updated as test data is processed.

## Acknowledgments

We thank Patrick Nguyen for helpful discussions regarding maximum entropy modeling and feature functions for handling structured and dynamic output classification problems.

## References

- AT&T Statistical Dialog Toolkit. AT&T Statistical Dialog Toolkit. <http://www2.research.att.com/sw/tools/asdt/>, 2013.
- Adam Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71, 1996.

- Alan W. Black, S. Burger, B. Langner, G. Parment, and M. Eskenazi. Spoken dialog challenge 2010. In *Proc. of Workshop on Spoken Language Technologies (SLT)*, 2010.
- Dan Bohus and Alex Rudnicky. A k hypotheses + other belief updating model. In *Proc. of AAAI Workshop on Statistical and Empirical Approaches to Spoken Dialog Systems*, 2006.
- Milica Gašić and Steve Young. Effective handling of dialogue state in the hidden information state pomdp dialogue manager. *ACM Transactions on Speech and Language Processing*, 7, 2011.
- Eric Horvitz and Tim Paek. A computational architecture for conversation. In *Proc. of the 7th Intl. Conf. on User Modeling*, 1999.
- Allan H Murphy. A new vector partition of the probability score. *Journal of Applied Meteorology*, 12:595–600, 1973.
- Blaise Thomson and Steve Young. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech and Language*, 24(4):562–588, 2010.
- Jason D. Williams. Incremental partition recombination for efficient tracking of multiple dialogue states. In *Proc. of ICASSP*, 2010.
- Jason D. Williams. Challenges and opportunities for state tracking in statistical spoken dialog systems: Results from two public deployments. *IEEE Journal of Selected Topics in Signal Processing, Special Issue on Advances in Spoken Dialogue Systems and Mobile Interface*, 6(8): 959–970, 2012.
- Jason D. Williams and Suhril Balakrishnan. Estimating probability of correctness for asr n-best lists. In *Proc. SigDial Conference*, 2009.
- Jason D. Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21:393–422, 2007.
- Jason D. Williams, Iker Arizmendi, and Alistair Conkie. Demonstration of AT&T Let’s Go: A production-grade statistical spoken dialog system. In *Proc of Workshop on Spoken Language Technologies (SLT)*, 2010.
- Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. The hidden information state model: a practical framework for POMDP-based spoken dialogue management. *Computer Speech and Language*, 24(2):150–174, 2010.
- Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proc. of the eighth ACM SIGKDD Intl. Conf on Knowledge Discovery and Data mining*, pages 694–699, 2002.

# Leveraging Synthetic Discourse Data via Multi-task Learning for Implicit Discourse Relation Recognition

Man Lan and Yu Xu

Department of Computer Science and Technology  
East China Normal University  
Shanghai, P.R.China  
mlan@cs.ecnu.edu.cn  
51101201049@ecnu.cn

Zheng-Yu Niu

Baidu Inc.  
Beijing, P.R.China  
niuzhengyu@baidu.com

## Abstract

To overcome the shortage of labeled data for implicit discourse relation recognition, previous works attempted to automatically generate training data by removing explicit discourse connectives from sentences and then built models on these synthetic implicit examples. However, a previous study (Sporleder and Lascarides, 2008) showed that models trained on these synthetic data do not generalize very well to natural (i.e. *genuine*) implicit discourse data. In this work we revisit this issue and present a multi-task learning based system which can effectively use synthetic data for implicit discourse relation recognition. Results on PDTB data show that under the multi-task learning framework our models with the use of the prediction of explicit discourse connectives as auxiliary learning tasks, can achieve an averaged  $F_1$  improvement of 5.86% over baseline models.

## 1 Introduction

The task of implicit discourse relation recognition is to identify the type of discourse relation (a.k.a. *rhetorical relation*) hold between two spans of text, where there is no discourse connective (a.k.a. *discourse marker*, e.g., *but*, *and*) in context to explicitly mark their discourse relation (e.g., *Contrast* or *Explanation*). It can be of great benefit to many downstream NLP applications, such as question answering (QA) (Verberne et al., 2007), information extraction (IE) (Cimiano et al., 2005), and machine translation (MT), etc. This task is quite challenging due to two reasons. First, without discourse connective in text, the task is quite difficult in itself. Second, implicit discourse relation is quite frequent in text. For example, almost half the sentences in the British National Corpus

held implicit discourse relations (Sporleder and Lascarides, 2008). Therefore, the task of implicit discourse relation recognition is the key to improving end-to-end discourse parser performance.

To overcome the shortage of manually annotated training data, (Marcu and Echihiabi, 2002) proposed a pattern-based approach to automatically generate training data from raw corpora. This line of research was followed by (Sporleder and Lascarides, 2008) and (Blair-Goldensohn, 2007). In these works, sentences containing certain words or phrases (e.g. *but*, *although*) were selected out from raw corpora using a pattern-based approach and then these words or phrases were removed from these sentences. Thus the resulting sentences were used as synthetic training examples for implicit discourse relation recognition. Since there is ambiguity of a word or phrase serving for discourse connective (i.e., the ambiguity between discourse and non-discourse usage or the ambiguity between two or more discourse relations if the word or phrase is used as a discourse connective), the synthetic implicit data would contain a lot of noises. Later, with the release of manually annotated corpus, such as Penn Discourse Treebank 2.0 (PDTB) (Prasad et al., 2008), recent studies performed implicit discourse relation recognition on natural (i.e., *genuine*) implicit discourse data (Pitler et al., 2009) (Lin et al., 2009) (Wang et al., 2010) with the use of linguistically informed features and machine learning algorithms.

(Sporleder and Lascarides, 2008) conducted a study of the pattern-based approach presented by (Marcu and Echihiabi, 2002) and showed that the model built on synthetic implicit data has not generalize well on natural implicit data. They found some evidence that this behavior is largely independent of the classifiers used and seems to lie in the data itself (e.g., marked and unmarked examples may be too dissimilar linguistically and

removing unambiguous markers in the automatic labelling process may lead to a meaning shift in the examples). We state that in some cases it is true while in other cases it may not always be so. A simple example is given here:

- (E1) a. We can't win.  
b. [*but*] We must keep trying.

We may find that in this example whether the insertion or the removal of connective *but* would not lead to a redundant or missing information between the above two sentences. That is, discourse connectives can be inserted between or removed from two sentences without changing the semantic relations between them in some cases. Another similar observation is in the annotation procedure of PDTB. To label implicit discourse relation, annotators inserted connective which can best express the relation between sentences without any redundancy<sup>1</sup>. We see that there should be some linguistic similarities between explicit and implicit discourse examples. Therefore, the first question arises: can we exploit this kind of linguistic similarity between explicit and implicit discourse examples to improve implicit discourse relation recognition?

In this paper, we propose a multi-task learning based method to improve the performance of implicit discourse relation recognition (as main task) with the help of relevant auxiliary tasks. Specifically, the main task is to recognize the implicit discourse relations based on genuine implicit discourse data and the auxiliary task is to recognize the implicit discourse relations based on synthetic implicit discourse data. According to the principle of multi-task learning, the learning model can be optimized by the shared part of the main task and the auxiliary tasks without bring unnecessary noise. That means, the model can learn from synthetic implicit data while it would not bring unnecessary noise from synthetic implicit data.

Although (Sporleder and Lascarides, 2008) did not mention, we speculate that another possible reason for the reported worse performance may result from noises in synthetic implicit discourse data. These synthetic data can be generated from two sources: (1) raw corpora with the use of pattern-based approach in (Marcu and Echihiabi,

<sup>1</sup>According to the PDTB Annotation Manual (PDTB-Group, 2008), if the insertion of connective leads to "redundancy", the relation is annotated as Alternative lexicalizations (AltLex), not implicit.

2002) and (Sporleder and Lascarides, 2008), and (2) manually annotated explicit data with the removal of explicit discourse connectives. Obviously, the data generated from the second source is cleaner and more reliable than that from the first source. Therefore, the second question to address in this work is: whether synthetic implicit discourse data generated from explicit discourse data source (i.e., the second source) can lead to a better performance than that from raw corpora (i.e., the first source)? To answer this question, we will make a comparison of synthetic discourse data generated from two corpora, i.e., the BILLIP corpus and the explicit discourse data annotated in PDTB.

The rest of this paper is organized as follows. Section 2 reviews related work on implicit discourse relation classification and multi-task learning. Section 3 presents our proposed multi-task learning method for implicit discourse relation classification. Section 4 provides the implementation technique details of the proposed multi-task method. Section 5 presents experiments and discusses results. Section 6 concludes this work.

## 2 Related Work

### 2.1 Implicit discourse relation classification

#### 2.1.1 Unsupervised approaches

Due to the lack of benchmark data for implicit discourse relation analysis, earlier work used unlabeled data to generate synthetic implicit discourse data. For example, (Marcu and Echihiabi, 2002) proposed an unsupervised method to recognize four discourse relations, i.e., *Contrast*, *Explanation-evidence*, *Condition* and *Elaboration*. They first used unambiguous pattern to extract explicit discourse examples from raw corpus. Then they generated synthetic implicit discourse data by removing explicit discourse connectives from sentences extracted. In their work, they collected word pairs from synthetic data set as features and used machine learning method to classify implicit discourse relation. Based on this work, several researchers have extended the work to improve the performance of relation classification. For example, (Saito et al., 2006) showed that the use of phrasal patterns as additional features can help a word-pair based system for discourse relation prediction on a Japanese corpus. Furthermore, (Blair-Goldensohn, 2007) improved previous work with the use of parameter optimization,

topic segmentation and syntactic parsing. However, (Sporleder and Lascarides, 2008) showed that the training model built on a synthetic data set, like the work of (Marcu and Echihiabi, 2002), may not be a good strategy since the linguistic dissimilarity between explicit and implicit data may hurt the performance of a model on natural data when being trained on synthetic data.

### 2.1.2 Supervised approaches

This line of research work approaches this relation prediction problem by recasting it as a classification problem. (Soricut and Marcu, 2003) parsed the discourse structures of sentences on RST Bank data set (Carlson et al., 2001) which is annotated based on Rhetorical Structure Theory (Mann and Thompson, 1988). (Wellner et al., 2006) presented a study of discourse relation disambiguation on GraphBank (Wolf et al., 2005). Recently, (Pitler et al., 2009) (Lin et al., 2009) and (Wang et al., 2010) conducted discourse relation study on PDTB (Prasad et al., 2008) which has been widely used in this field.

### 2.1.3 Semi-supervised approaches

Research work in this category exploited both labeled and unlabeled data for discourse relation prediction. (Hernault et al., 2010) presented a semi-supervised method based on the analysis of co-occurring features in labeled and unlabeled data. Very recently, (Hernault et al., 2011) introduced a semi-supervised work using structure learning method for discourse relation classification, which is quite relevant to our work. However, they performed discourse relation classification on both explicit and implicit data. And their work is different from our work in many aspects, such as, feature sets, auxiliary task, auxiliary data, class labels, learning framework, and so on. Furthermore, there is no explicit conclusion or evidence in their work to address the two questions raised in Section 1.

Unlike their previous work, our previous work (Zhou et al., 2010) presented a method to predict the missing connective based on a language model trained on an unannotated corpus. The predicted connective was then used as a feature to classify the implicit relation.

## 2.2 Multi-task learning

Multi-task learning is a kind of machine learning method, which learns a main task together with

other related auxiliary tasks at the same time, using a shared representation. This often leads to a better model for the main task, because it allows the learner to use the commonality among the tasks. Many multi-task learning methods have been proposed in recent years, (Ando and Zhang, 2005a), (Argyriou et al., 2008), (Jebara, 2004), (Bonilla et al., 2008), (Evgeniou and Pontil, 2004), (Baxter, 2000), (Caruana, 1997), (Thrun, 1996). One group uses task relations as regularization terms in the objective function to be optimized. For example, in (Evgeniou and Pontil, 2004) the regularization terms make the parameters of models closer for similar tasks. Another group is proposed to find the common structure from data and then utilize the learned structure for multi-task learning (Argyriou et al., 2008) (Ando and Zhang, 2005b).

## 3 Multi-task Learning for Discourse Relation Prediction

### 3.1 Motivation

The idea of using multi-task learning for implicit discourse relation classification is motivated by the observations that we have made on implicit discourse relation.

On one hand, since building a hand-annotated implicit discourse relation corpus is costly and time consuming, most previous work attempted to use synthetic implicit discourse examples as training data. However, (Sporleder and Lascarides, 2008) found that the model trained on synthetic implicit data has not performed as well as expected in natural implicit data. They stated that the reason is linguistic dissimilarity between explicit and implicit discourse data. This indicates that straightly using synthetic implicit data as training data may not be helpful.

On the other hand, as shown in Section 1, we observe that in some cases explicit discourse relation and implicit discourse relation can express the same meaning with or without a discourse connective. This indicates that in certain degree they must be similar to each other. If it is true, the synthetic implicit relations are expected to be helpful for implicit discourse relation classification. Therefore, what we have to do is to find a way to train a model which has the capabilities to learn from their similarity and to ignore their dissimilarity as well.

To solve it, we propose a multi-task learning method for implicit discourse relation classi-

fication, where the classification model seeks the shared part through jointly learning main task and multiple auxiliary tasks. As a result, the model can be optimized by the similar shared part without bringing noise in the dissimilar part. Specifically, in this work, we use alternating structure optimization (ASO) (Ando and Zhang, 2005a) to construct the multi-task learning framework. ASO has been shown to be useful in a *semi-supervised learning* configuration for several NLP applications, such as, text chunking (Ando and Zhang, 2005b) and text classification (Ando and Zhang, 2005a).

### 3.2 Multi-task learning and ASO

Generally, multi-task learning (MTL) considers  $m$  prediction problems indexed by  $\ell \in \{1, \dots, m\}$ , each with  $n_\ell$  samples  $(X_i^\ell, Y_i^\ell)$  for  $i \in \{1, \dots, n_\ell\}$  ( $X_i$  are input *feature vectors* and  $Y_i$  are corresponding *classification labels*) and assumes that there exists a common predictive structure shared by these  $m$  problems. Generally, the joint linear model for MTL is to predict problem  $\ell$  in the following form:

$$f_\ell(\Theta, X) = w_\ell^T X + v_\ell^T \Theta X, \Theta \Theta^T = I, \quad (1)$$

where  $I$  is the identity matrix,  $w_\ell$  and  $v_\ell$  are weight vectors specific to each problem  $\ell$ , and  $\Theta$  is the structure matrix shared by all the  $m$  predictors. The main goal of MTL is to learn a common good feature map  $\Theta X$  for all the  $m$  problems. Several MTL methods have been presented to learn  $\Theta X$  for all the  $m$  problems. In this work, we adopt the ASO method.

Specifically, the ASO method adopted *singular value decomposition* (SVD) to obtain  $\Theta$  and  $m$  predictors that minimize the empirical risk summed over all the  $m$  problems. Thus, the problem of optimization becomes the minimization of the joint empirical risk written as:

$$\sum_{\ell=1}^m \left( \sum_{i=1}^{n_\ell} \frac{L(f_\ell(\Theta, X_i^\ell), Y_i)}{n_\ell} + \lambda \|W_\ell\|^2 \right) \quad (2)$$

where loss function  $L(\cdot)$  quantifies the difference between the prediction  $f(X_i)$  and the true output  $Y_i$  for each predictor, and  $\lambda$  is a regularization parameter for square regularization to control the model complexity. To minimize the empirical risk, ASO repeats the following alternating optimization procedure until a convergence criterion is met:

- 1) Fix  $(\Theta, V_\ell)$ , and find  $m$  predictors  $f_\ell$  that minimize the above joint empirical risk.
- 2) Fix  $m$  predictors  $f_\ell$ , and find  $(\Theta, V_\ell)$  that minimizes the above joint empirical risk.

### 3.3 Auxiliary tasks

There are two main principles to create auxiliary tasks. First, the auxiliary tasks should be automatically labeled in order to reduce the cost of manual labeling. Second, since the MTL model learns from the shared part of main task and auxiliary tasks, the auxiliary tasks should be quite relevant/similar to the main task. It is generally believed that the more the auxiliary tasks are relevant to the main task, the more the main task can benefit from the auxiliary tasks. Following these two principles, we create the auxiliary tasks by generating automatically labeled data as follows.

Previous work (Marcu and Echiabi, 2002) and (Sporleder and Lascarides, 2008) adopted predefined pattern-based approach to generate synthetic labeled data, where each predefined pattern has one discourse relation label. In contrast, we adopt an automatic approach to generate synthetic labeled data, where each discourse connective between two texts serves as their relation label. The reason lies in the very strong connection between discourse connectives and discourse relations. For example, the connective *but* always indicates a *contrast* relation between two texts. And (Pitler et al., 2008) proved that using only connective itself, the accuracy of explicit discourse relation classification is over 93%.

To build the mapping between discourse connective and discourse relation, for each connective, we count the times it appears in each relation and regard the relation in which it appears most frequently as its most relevant relation. Based on this mapping between connective and relation, we extract the synthetic labeled data containing the connective as training data for auxiliary tasks.

For example, *and* appears 3,000 times in PDTB as a discourse connective. Among them, it is manually annotated as an *Expansion* relation for 2,938 times. So we regard the *Expansion* relation as its most relevant relation and generate a mapping pattern like: “*and*  $\rightarrow$  *Expansion*”. Then we extract all sentences which contain discourse “*and*” and remove this connective “*and*” from sentences to generate synthetic implicit data. The resulting sentences are used in auxiliary task and automatically

marked as *Expansion* relation.

## 4 Implementation Details of Multi-task Learning Method

### 4.1 Data sets for *main* and *auxiliary* tasks

To examine whether there is a difference in synthetic implicit data generated from unannotated and annotated corpus, we use two corpora. One is a hand-annotated explicit discourse corpus, i.e., the explicit discourse relations in PDTB, denoted as *exp*. Another is an unannotated corpus, i.e., BLLIP (David McClosky and Johnson., 2008).

#### 4.1.1 Penn Discourse Treebank

PDTB (Prasad et al., 2008) is the largest hand-annotated corpus of discourse relation so far. It contains 2,312 Wall Street Journal (WSJ) articles. The sense label of discourse relations is hierarchically with three levels, i.e., *class*, *type* and *subtype*. The top level contains four major semantic *classes*: *Comparison* (denoted as *Comp.*), *Contingency* (*Cont.*), *Expansion* (*Exp.*) and *Temporal* (*Temp.*). For each class, a set of *types* is used to refine relation sense. The set of *subtypes* is to further specify the semantic contribution of each argument. In this paper, we focus on the top level (*class*) and the second level (*type*) relations because the *subtype* relations are too fine-grained and only appear in some relations.

Both explicit and implicit discourse relations are labeled in PDTB. In our experiment, the implicit discourse relations are used in the main task and for evaluation. While the explicit discourse relations are used in the auxiliary task. A detailed description of the data sources for different tasks is given below.

**Data set for main task** Following previous work in (Pitler et al., 2009) and (Zhou et al., 2010), the implicit relations in sections 2-20 are used as training data for the main task (denoted as *imp*) and the implicit relations in sections 21-22 are for evaluation. Table 1 shows the distribution of implicit relations. There are too few training instances for six second level relations (indicated by \* in Table 1), so we removed these six relations in our experiments.

**Data set for auxiliary task** All explicit instances in sections 00-24 in PDTB, i.e., 18,459 instances, are used for auxiliary task (denoted as *exp*). Following the method described in Section 3.3, we build the mapping patterns between con-

Top level	Second level	train	test
Temp		736	83
	<i>Synchrony</i>	203	28
	<i>Asynchronous</i>	532	55
Cont		3333	279
	<i>Cause</i>	3270	272
	<i>Pragmatic Cause*</i>	64	7
	<i>Condition*</i>	1	0
	<i>Pragmatic condition*</i>	1	0
Comp		1939	152
	<i>Contrast</i>	1607	134
	<i>Pragmatic contrast*</i>	4	0
	<i>Concession</i>	183	17
	<i>Pragmatic concession*</i>	1	0
Exp		6316	567
	<i>Conjunction</i>	2872	208
	<i>Instantiation</i>	1063	119
	<i>Restatement</i>	2405	213
	<i>Alternative</i>	147	9
	<i>Exception*</i>	0	0
	<i>List</i>	338	12

Table 1: Distribution of implicit discourse relations in the top and second level of PDTB

nectives and relations in PDTB and generate synthetic labeled data by removing the connectives. According to the most relevant relation sense of connective removed, the resulting instances are grouped into different data sets.

#### 4.1.2 BLLIP

BLLIP North American News Text (Complete) is used as unlabeled data source to generate synthetic labeled data. In comparison with the synthetic labeled data generated from the explicit relations in PDTB, the synthetic labeled data from BLLIP contains more noise. This is because the former data is manually annotated whether a word serves as discourse connective or not, while the latter does not manually disambiguate two types of ambiguity, i.e., whether a word serves as discourse connective or not, and the type of discourse relation if it is a discourse connective. Finally, we extract 26,412 instances from BLLIP (denoted as *BLLIP*) and use them for auxiliary task.

### 4.2 Feature representation

For both main task and auxiliary tasks, we adopt the following three feature types. These features are chosen due to their superior performance in previous work (Pitler et al., 2009) and our previous work (Zhou et al., 2010).

**Verbs:** Following (Pitler et al., 2009), we extract the pairs of verbs from both text spans. The number of verb pairs which have the same highest

Levin verb class levels (Levin, 1993) is counted as a feature. Besides, the average length of verb phrases in each argument is included as a feature. In addition, the part of speech tags of the main verbs (e.g., base form, past tense, 3rd person singular present, etc.) in each argument, i.e., MD, VB, VBD, VBG, VBN, VBP, VBZ, are recorded as features, where we simply use the first verb in each argument as the main verb.

**Polarity:** This feature records the number of positive, negated positive, negative and neutral words in both arguments and their cross product as well. For negated positives, we first locate the negated words in text span and then define the closely behind positive word as negated positive. The polarity of each word in arguments is derived from Multi-perspective Question Answering Opinion Corpus (MPQA) (Wilson et al., 2009).

**Modality:** We examine six modal words (i.e., *can*, *may*, *must*, *need*, *shall*, *will*) including their various tenses or abbreviation forms in both arguments. This feature records the presence or absence of modal words in both arguments and their cross product.

### 4.3 Classifiers used multi-task learning

We extract the above linguistically informed features from two synthetic implicit data sets (i.e., *BLLIP* and *exp*) to learn the auxiliary classifier and from the natural implicit data set (i.e., *imp*) to learn the main classifier. Under the ASO-based multi-task learning framework, the model of main task learns from the shared part of main task and auxiliary tasks. Specifically, we adopt multiple binary classification to build model for main task. That is, for each discourse relation, we build a binary classifier.

## 5 Experiments and Results

### 5.1 Experiments

Although previous work has been done on PDTB (Pitler et al., 2009) and (Lin et al., 2009), we cannot make a direct comparison with them because various experimental conditions, such as, different classification strategies (multi-class classification, multiple binary classification), different data preparation (feature extraction and selection), different benchmark data collections (different sections for training and test, different levels of discourse relations), different classifiers with various parameters (MaxEnt, Naïve Bayes, SVM, etc) and

even different evaluation methods ( $F_1$ , accuracy) have been adopted by different researchers.

Therefore, to address the two questions raised in Section 1 and to make the comparison reliable and reasonable, we performed experiments on the top and second level of PDTB using single task learning and multi-task learning, respectively. The systems using single task learning serve as baseline systems. Under the single task learning, various combinations of *exp* and *BLLIP* data are incorporated with *imp* data for the implicit discourse relation classification task.

We hypothesize that synthetic implicit data would contribute to the main task, i.e., the implicit discourse relation classification. Specifically, the natural implicit data (i.e., *imp*) are used to create main task and the synthetic implicit data (*exp* or *BLLIP*) are used to create auxiliary tasks for the purpose of optimizing the objective functions of main task. If the hypothesis is correct, the performance of main task would be improved by auxiliary tasks created from synthetic implicit data. Thus in the experiments of multi-task learning, only natural implicit examples (i.e., *imp*) data are used for main task training while different combinations of synthetic implicit examples (*exp* and *BLLIP*) are used for auxiliary task training.

We adopt precision, recall and their combination  $F_1$  for performance evaluation. We also perform one-tailed t-test to validate if there is significant difference between two methods in terms of  $F_1$  performance analysis.

### 5.2 Results

Table 2 summarizes the experimental results under single and multi-task learning on the top level of four PDTB relations with respect to different combinations of synthetic implicit data. For each relation, the first three rows indicate the results of using different single training data under single task learning and the last three rows indicate the results using different combinations of training data under single task and multi-task learning. The best  $F_1$  for every relation is shown in bold font. From this table, we can find that on four relations, our multi-task learning systems achieved the best performance using the combination of *exp* and *BLLIP* synthetic data.

Table 3 summarizes the best single task and the best multi-task learning results on the second level of PDTB. For four relations, i.e., Synchrony, Con-

Level 1 class	Single-task				Multi-task				
	Data	P	R	$F_1$	Data (main)	Data (aux)	P	R	$F_1$
Comp.	<i>imp</i>	21.43	37.50	27.27	-	-	-	-	-
	<i>BLLIP</i>	12.68	53.29	20.48	-	-	-	-	-
	<i>exp</i>	15.25	50.66	23.44	-	-	-	-	-
	<i>imp + exp</i>	16.94	40.13	23.83	<i>imp</i>	<i>exp</i>	22.94	49.34	30.90
	<i>imp + BLLIP</i>	13.56	44.08	20.74	<i>imp</i>	<i>BLLIP</i>	20.47	63.16	30.92
	<i>imp + exp + BLLIP</i>	14.54	38.16	21.05	<i>imp</i>	<i>exp + BLLIP</i>	23.47	48.03	<b>31.53</b>
Cont.	<i>imp</i>	37.65	43.73	40.46	-	-	-	-	-
	<i>BLLIP</i>	33.72	31.18	32.40	-	-	-	-	-
	<i>exp</i>	35.24	26.52	30.27	-	-	-	-	-
	<i>imp + exp</i>	39.00	13.98	20.58	<i>imp</i>	<i>exp</i>	39.94	45.52	42.55
	<i>imp + BLLIP</i>	37.30	24.73	29.74	<i>imp</i>	<i>BLLIP</i>	37.80	63.80	47.47
	<i>imp + exp + BLLIP</i>	39.37	31.18	34.80	<i>imp</i>	<i>exp + BLLIP</i>	35.90	70.25	<b>47.52</b>
Exp.	<i>imp</i>	56.59	66.67	61.21	-	-	-	-	-
	<i>BLLIP</i>	53.29	40.04	45.72	-	-	-	-	-
	<i>exp</i>	57.97	58.38	58.17	-	-	-	-	-
	<i>imp + exp</i>	57.32	65.61	61.18	<i>imp</i>	<i>exp</i>	59.14	67.90	63.22
	<i>imp + BLLIP</i>	56.28	65.61	60.59	<i>imp</i>	<i>BLLIP</i>	53.80	99.82	69.92
	<i>imp + exp + BLLIP</i>	55.81	65.26	60.16	<i>imp</i>	<i>exp + BLLIP</i>	53.90	99.82	<b>70.01</b>
Temp.	<i>imp</i>	16.46	63.86	26.17	-	-	-	-	-
	<i>BLLIP</i>	17.31	43.37	24.74	-	-	-	-	-
	<i>exp</i>	15.46	36.14	21.66	-	-	-	-	-
	<i>imp + exp</i>	15.35	39.76	22.15	<i>imp</i>	<i>exp</i>	18.60	63.86	28.80
	<i>imp + BLLIP</i>	14.74	33.73	20.51	<i>imp</i>	<i>BLLIP</i>	18.12	67.47	28.57
	<i>imp + exp + BLLIP</i>	15.94	39.76	22.76	<i>imp</i>	<i>exp + BLLIP</i>	19.08	65.06	<b>29.51</b>

Table 2: Performance of precision, recall and  $F_1$  for 4 Level 1 relation classes. “-” indicates N.A.

Level 2 type	Single-task				Multi-task				
	Data	P	R	$F_1$	Data (main)	Data (aux)	P	R	$F_1$
Asynchronous	<i>imp</i>	11.36	74.55	19.71	<i>imp</i>	<i>exp + BLLIP</i>	23.08	21.82	<b>22.43</b>
Synchrony	<i>imp</i>	-	-	-	<i>imp</i>	<i>exp + BLLIP</i>	-	-	-
Cause	<i>imp</i>	36.38	64.34	46.48	<i>imp</i>	<i>exp + BLLIP</i>	36.01	67.65	<b>47.00</b>
Contrast	<i>imp</i>	20.07	42.54	27.27	<i>imp</i>	<i>exp + BLLIP</i>	20.70	52.99	<b>29.77</b>
Concession	<i>imp</i>	-	-	-	<i>imp</i>	<i>exp + BLLIP</i>	-	-	-
Conjunction	<i>imp</i>	26.35	63.46	37.24	<i>imp</i>	<i>exp + BLLIP</i>	26.29	73.56	<b>38.73</b>
Instantiation	<i>imp</i>	22.78	53.78	32.00	<i>imp</i>	<i>exp + BLLIP</i>	22.55	57.98	<b>32.47</b>
Restatement	<i>imp</i>	23.11	67.61	34.45	<i>imp</i>	<i>exp + BLLIP</i>	26.93	53.99	<b>35.94</b>
Alternative	<i>imp</i>	-	-	-	<i>imp</i>	<i>exp + BLLIP</i>	-	-	-
List	<i>imp</i>	-	-	-	<i>imp</i>	<i>exp + BLLIP</i>	-	-	-

Table 3: Performance of precision, recall and  $F_1$  for 10 Level 2 relation types. “-” indicates 0.00.

cession, Alternative and List, the classifier labels no instances due to the small percentages for these four types.

Table 4 summarizes the one-tailed t-test results on the top level of PDTB between the best single task learning system (i.e., *imp*) and three multi-task learning systems (*imp:exp+BLLIP* indicates that *imp* is used for main task and the combination of *exp* and *BLLIP* are for auxiliary task). The systems with insignificant performance differences are grouped into one set and “>” and “>>” denote better than at significance level 0.01 and

0.001 respectively.

### 5.3 Discussion

From Table 2 to Table 4, several findings can be found as follows.

We can see that the multi-task learning systems perform consistently better than the single task learning systems for the prediction of implicit discourse relations. Our best multi-task learning system achieves an averaged  $F_1$  improvement of 5.86% over the best single task learning system on the top level of PDTB relations. Specifically, for

Class	One-tailed t-test results
Comp.	$(imp:exp+BLLIP, imp:exp, imp:BLLIP) \gg (imp)$
Cont.	$(imp:exp+BLLIP, imp:BLLIP) \gg (imp:exp) > (imp)$
Exp.	$(imp:exp+BLLIP, imp:BLLIP) \gg (imp:exp) > (imp)$
Temp.	$(imp:exp+BLLIP, imp:exp, imp:BLLIP) \gg (imp)$

Table 4: Statistical significance tests results.

the relations *Comp.*, *Cont.*, *Exp.*, *Temp.*, our best multi-task learning system achieve 4.26%, 7.06%, 8.8% and 3.34%  $F_1$  improvements over the best single task learning system. It indicates that using synthetic implicit data as auxiliary task greatly improves the performance of the main task. This is confirmed by the following t-tests in Table 4.

In contrast to the performance of multi-task learning, the performance of the best single task learning system has been achieved on natural implicit discourse data alone. This finding is consistent with (Sporleder and Lascarides, 2008). It indicates that under single task learning, directly adding synthetic implicit data to increase the number of training data cannot be helpful to implicit discourse relation classification. The possible reasons result from (1) the different nature of implicit and explicit discourse data in linguistics and (2) the noise brought from synthetic implicit data.

Based on the above analysis, we state that it is the way of utilizing synthetic implicit data that is important for implicit discourse relation classification.

Although all three multi-task learning systems outperformed single task learning systems, we find that the two synthetic implicit data sets have not been shown a universally consistent performance on four top level PDTB relations. On one hand, for the relations *Comp.* and *Temp.*, the performance of the two synthetic implicit data sets alone and their combination are comparable to each other and there is no significant difference between them. On the other hand, for the relations *Cont.* and *Exp.*, the performance of *exp* data is inferior to that of *BLLIP* and their combination. This is contrary to our original expectation that *exp* data which has been manually annotated for discourse connective disambiguation should outperform *BLLIP* which contains a lot of noise. This finding indicates that under the multi-task learning, it may not be worthy of using manually annotated corpus to generate auxiliary data. It is quite promising since it can provide benefits to reducing

the cost of human efforts on corpus annotation.

## 5.4 Ambiguity Analysis

Although our experiments show that synthetic implicit data can help implicit discourse relation classification under multi-task learning framework, the overall performance is still quite low (44.64% in  $F_1$ ). Therefore, we analyze the types of ambiguity in relations and connectives in order to motivate possible future work.

### 5.4.1 Ambiguity of implicit relation

Without explicit discourse connective, the implicit discourse relation instance can be understood in two or more different ways. Given the example E2 in PDTB, the PDTB annotators explain it as *Contingency* or *Expansion* relation and manually insert corresponding implicit connective *for one thing* or *because* to express its relation.

- (E2) **Arg1:**Now the stage is set for the battle to play out  
**Arg2:**The anti-programmers are getting some helpful thunder from Congress  
**Connective1:**because  
**Sense1:**Contingency.Cause.Reason  
**Connective2:**for one thing  
**Sense2:**Expansion.Instantiation (wsj\_0118)

Thus the ambiguity of implicit discourse relations makes this task difficult in itself.

### 5.4.2 Ambiguity of discourse connectives

As we mentioned before, even given an explicit discourse connective in text, its discourse relation still can be explained in two or more different ways. And for different connectives, the ambiguity of relation senses is quite different. That is, the most frequent sense is not always the only sense that a connective expresses. In example E3, “*since*” is explained by annotators to express *Temporal* or *Contingency* relation.

- (E3) **Arg1:**MiniScribe has been on the rocks  
**Arg2:****since** it disclosed early this year that its earnings reports for 1988 weren’t accurate.

Sense1:Temporal.Asynchronous.Succession

Sense2:Contingency.Cause.Reason

(wsj\_0003)

In PDTB, “since” appears 184 times in explicit discourse relations. It expresses *Temporal* relation for 80 times, *Contingency* relation for 94 times and both *Temporal* and *Contingency* for 10 time (like example E3). Therefore, although we use its most frequent sense, i.e., *Contingency*, to automatically extract sentences and label them, almost less than half of them actually express *Temporal* relation. Thus the ambiguity of discourse connectives is another source which has brought noise to data when we generate synthetical implicit discourse relation.

## 6 Conclusions

In this paper, we present a multi-task learning method to improve implicit discourse relation classification by leveraging synthetic implicit discourse data. Results on PDTB show that under the framework of multi-task learning, using synthetic discourse data as auxiliary task significantly improves the performance of main task. Our best multi-task learning system achieves an averaged  $F_1$  improvement of 5.86% over the best single task learning system on the top level of PDTB relations. Specifically, for the relations *Comp.*, *Cont.*, *Exp.*, *Temp.*, our best multi-task learning system achieves 4.26%, 7.06%, 8.8%, and 3.34%  $F_1$  improvements over a state of the art baseline system. This indicates that it is the way of utilizing synthetic discourse examples that is important for implicit discourse relation classification.

## Acknowledgements

This research is supported by grants from National Natural Science Foundation of China (No.60903093), Shanghai Pujiang Talent Program (No.09PJ1404500), Doctoral Fund of Ministry of Education of China (No. 20090076120029) and Shanghai Knowledge Service Platform Project (No. ZF1213).

## References

- R.K. Ando and T. Zhang. 2005a. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853.
- R.K. Ando and T. Zhang. 2005b. A high-performance semi-supervised learning method for text chunking.

pages 1–9. Association for Computational Linguistics. Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics.

- A. Argyriou, C.A. Micchelli, M. Pontil, and Y. Ying. 2008. A spectral regularization framework for multi-task structure learning. *Advances in Neural Information Processing Systems*, 20:2532.
- J. Baxter. 2000. A model of inductive bias learning. *J. Artif. Intell. Res. (JAIR)*, 12:149–198.
- S.J. Blair-Goldensohn. 2007. *Long-answer question answering and rhetorical-semantic relations*. Ph.D. thesis.
- E. Bonilla, K.M. Chai, and C. Williams. 2008. Multi-task gaussian process prediction. *Advances in Neural Information Processing Systems*, 20(October).
- L. Carlson, D. Marcu, and M.E. Okunowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. pages 1–10. Association for Computational Linguistics. Proceedings of the Second SIGdial Workshop on Discourse and Dialogue-Volume 16.
- R. Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.
- P. Cimiano, U. Reyle, and J. Saric. 2005. Ontology-driven discourse analysis for information extraction. *Data and Knowledge Engineering*, 55(1):59–83.
- Eugene Charniak David McClosky and Mark Johnson. 2008. Bllip north american news text, complete.
- T. Evgeniou and M. Pontil. 2004. Regularized multi-task learning. pages 109–117. ACM. Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining.
- H. Hernault, D. Bollegala, and M. Ishizuka. 2010. A semi-supervised approach to improve classification of infrequent discourse relations using feature vector extension. pages 399–409. Association for Computational Linguistics. Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing.
- H. Hernault, D. Bollegala, and M. Ishizuka. 2011. Semi-supervised discourse relation classification with structural learning. In *Proceedings of the 12th international conference on Computational linguistics and intelligent text processing - Volume Part I*, CICLing’11, pages 340–352, Berlin, Heidelberg. Springer-Verlag.
- T. Jebara. 2004. Multi-task feature and kernel selection for svms. page 55. ACM. Proceedings of the twenty-first international conference on Machine learning.
- B. Levin. 1993. *English verb classes and alternations: A preliminary investigation*, volume 348. University of Chicago press Chicago, IL.

- Z. Lin, M.Y. Kan, and H.T. Ng. 2009. Recognizing implicit discourse relations in the penn discourse treebank. pages 343–351. Association for Computational Linguistics. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1.
- W.C. Mann and S.A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.
- D. Marcu and A. Echiabi. 2002. An unsupervised approach to recognizing discourse relations. pages 368–375. Association for Computational Linguistics. Proceedings of the 40th Annual Meeting on Association for Computational Linguistics.
- PDTB-Group. 2008. The penn discourse treebank 2.0 annotation manual. Technical report, Institute for Research in Cognitive Science, University of Pennsylvania.
- E. Pitler, M. Raghupathy, H. Mehta, A. Nenkova, A. Lee, and A. Joshi. 2008. Easily identifiable discourse relations. Citeseer. Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008), Manchester, UK, August.
- E. Pitler, A. Louis, and A. Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. pages 683–691. Association for Computational Linguistics. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of LREC*.
- M. Saito, K. Yamamoto, and S. Sekine. 2006. Using phrasal patterns to identify discourse relations. pages 133–136. Association for Computational Linguistics. Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers on XX.
- R. Soricut and D. Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. pages 149–156. Association for Computational Linguistics. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1.
- C. Sporleder and A. Lascarides. 2008. Using automatically labelled examples to classify rhetorical relations: An assessment. *Natural Language Engineering*, 14(03):369–416.
- S. Thrun. 1996. Is learning the n-th thing any easier than learning the first? *Advances in Neural Information Processing Systems*, pages 640–646.
- S. Verberne, L. Boves, N. Oostdijk, and P.A. Coppen. 2007. Evaluating discourse-based answer extraction for why-question answering. pages 735–736. ACM. Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval.
- W.T. Wang, J. Su, and C.L. Tan. 2010. Kernel based discourse relation recognition with temporal ordering information. pages 710–719. Association for Computational Linguistics. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics.
- B. Wellner, J. Pustejovsky, C. Havasi, A. Rumshisky, and R. Sauri. 2006. Classification of discourse coherence relations: An exploratory study using multiple knowledge sources. pages 117–125. Association for Computational Linguistics. Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue.
- T. Wilson, J. Wiebe, and P. Hoffmann. 2009. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational Linguistics*, 35(3):399–433.
- F. Wolf, E. Gibson, A. Fisher, and M. Knight. 2005. The discourse graphbank: A database of texts annotated with coherence relations. *Linguistic Data Consortium*.
- Z.M. Zhou, Y. Xu, Z.Y. Niu, M. Lan, J. Su, and C.L. Tan. 2010. Predicting discourse connectives for implicit discourse relation recognition. pages 1507–1514. Association for Computational Linguistics. Proceedings of the 23rd International Conference on Computational Linguistics: Posters.

# Combining Intra- and Multi-sentential Rhetorical Parsing for Document-level Discourse Analysis

Shafiq Joty\*

sjoty@qf.org.qa

Qatar Computing Research Institute  
Qatar Foundation  
Doha, Qatar

Giuseppe Carenini, Raymond Ng, Yashar Mehdad

{carenini, rng, mehdad}@cs.ubc.ca

Department of Computer Science  
University of British Columbia  
Vancouver, Canada

## Abstract

We propose a novel approach for developing a two-stage document-level discourse parser. Our parser builds a discourse tree by applying an optimal parsing algorithm to probabilities inferred from two Conditional Random Fields: one for intra-sentential parsing and the other for multi-sentential parsing. We present two approaches to combine these two stages of discourse parsing effectively. A set of empirical evaluations over two different datasets demonstrates that our discourse parser significantly outperforms the state-of-the-art, often by a wide margin.

## 1 Introduction

Discourse of any kind is not formed by independent and isolated textual units, but by related and structured units. Discourse analysis seeks to uncover such structures underneath the surface of the text, and has been shown to be beneficial for text summarization (Louis et al., 2010; Marcu, 2000b), sentence compression (Sporleder and Lapata, 2005), text generation (Prasad et al., 2005), sentiment analysis (Somasundaran, 2010) and question answering (Verberne et al., 2007).

Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), one of the most influential theories of discourse, represents texts by labeled hierarchical structures, called Discourse Trees (DTs), as exemplified by a sample DT in Figure 1. The leaves of a DT correspond to contiguous Elementary Discourse Units (EDUs) (six in the example). Adjacent EDUs are connected by rhetorical relations (e.g., *Elaboration*, *Contrast*), forming larger discourse units (represented by internal

nodes), which in turn are also subject to this relation linking. Discourse units linked by a rhetorical relation are further distinguished based on their relative importance in the text: *nucleus* being the central part, whereas *satellite* being the peripheral one. Discourse analysis in RST involves two sub-tasks: *discourse segmentation* is the task of identifying the EDUs, and *discourse parsing* is the task of linking the discourse units into a labeled tree.

While recent advances in automatic discourse segmentation and sentence-level discourse parsing have attained accuracies close to human performance (Fisher and Roark, 2007; Joty et al., 2012), discourse parsing at the document-level still poses significant challenges (Feng and Hirst, 2012) and the performance of the existing document-level parsers (Hernault et al., 2010; Subba and Di-Eugenio, 2009) is still considerably inferior compared to human gold-standard. This paper aims to reduce this performance gap and take discourse parsing one step further. To this end, we address three key limitations of existing parsers as follows.

First, existing discourse parsers typically model the structure and the labels of a DT separately in a *pipeline* fashion, and also do not consider the sequential dependencies between the DT constituents, which has been recently shown to be critical (Feng and Hirst, 2012). To address this limitation, as the first contribution, we propose a novel document-level discourse parser based on probabilistic discriminative parsing models, represented as Conditional Random Fields (CRFs) (Sutton et al., 2007), to infer the probability of all possible DT constituents. The CRF models effectively represent the structure and the label of a DT constituent jointly, and whenever possible, capture the sequential dependencies between the constituents.

Second, existing parsers apply greedy and sub-optimal parsing algorithms to build the DT for a document. To cope with this limitation, our CRF models support a probabilistic bottom-up parsing

---

This work was conducted at the University of British Columbia, Vancouver, Canada.

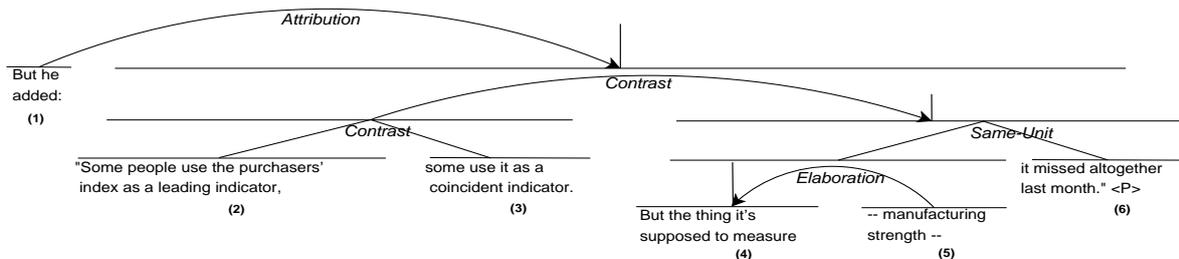


Figure 1: Discourse tree for two sentences in RST-DT. Each of the sentences contains three EDUs. The second sentence has a well-formed discourse tree, but the first sentence does not have one.

algorithm which is non-greedy and optimal.

Third, existing discourse parsers do not discriminate between intra-sentential (i.e., building the DTs for the individual sentences) and multi-sentential parsing (i.e., building the DT for the document). However, we argue that distinguishing between these two conditions can result in more effective parsing. Two separate parsing models could exploit the fact that rhetorical relations are distributed differently intra-sententially vs. multi-sententially. Also, they could independently choose their own informative features. As another key contribution of our work, we devise two different parsing components: one for intra-sentential parsing, the other for multi-sentential parsing. This provides for scalable, modular and flexible solutions, that can exploit the strong correlation observed between the text structure (sentence boundaries) and the structure of the DT.

In order to develop a complete and robust discourse parser, we combine our intra-sentential and multi-sentential parsers in two different ways. Since most sentences have a well-formed discourse sub-tree in the full document-level DT (for example, the second sentence in Figure 1), our first approach constructs a DT for every sentence using our intra-sentential parser, and then runs the multi-sentential parser on the resulting sentence-level DTs. However, this approach would disregard those cases where rhetorical structures violate sentence boundaries. For example, consider the first sentence in Figure 1. It does not have a well-formed sub-tree because the unit containing EDUs 2 and 3 merges with the next sentence and only then is the resulting unit merged with EDU 1. Our second approach, in an attempt of dealing with these cases, builds sentence-level sub-trees by applying the intra-sentential parser on a sliding window covering two adjacent sentences and by then consolidating the results produced by over-

lapping windows. After that, the multi-sentential parser takes all these sentence-level sub-trees and builds a full rhetorical parse for the document.

While previous approaches have been tested on only one corpus, we evaluate our approach on texts from two very different genres: news articles and instructional how-to-do manuals. The results demonstrate that our contributions provide consistent and statistically significant improvements over previous approaches. Our final result compares very favorably to the result of state-of-the-art models in document-level discourse parsing.

In the rest of the paper, after discussing related work in Section 2, we present our discourse parsing framework in Section 3. In Section 4, we describe the intra- and multi-sentential parsing components. Section 5 presents the two approaches to combine the two stages of parsing. The experiments and error analysis, followed by future directions are discussed in Section 6. Finally, we summarize our contributions in Section 7.

## 2 Related work

The idea of staging document-level discourse parsing on top of sentence-level discourse parsing was investigated in (Marcu, 2000a; LeThanh et al., 2004). These approaches mainly rely on discourse markers (or cues), and use hand-coded rules to build DTs for sentences first, then for paragraphs, and so on. However, often rhetorical relations are not explicitly signaled by discourse markers (Marcu and Echihiabi, 2002), and discourse structures do not always correspond to paragraph structures (Sporleder and Lascarides, 2004). Therefore, rather than relying on hand-coded rules based on discourse markers, recent approaches employ supervised machine learning techniques with a large set of informative features.

Hernault et al., (2010) presents the publicly available HILDA parser. Given the EDUs in a doc-

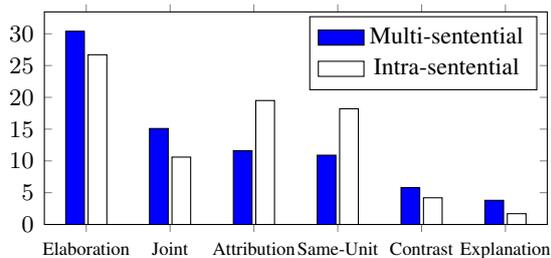


Figure 2: Distributions of six most frequent relations in intra-sentential and multi-sentential parsing scenarios.

ument, HILDA iteratively employs two Support Vector Machine (SVM) classifiers in pipeline to build the DT. In each iteration, a binary classifier first decides which of the adjacent units to merge, then a multi-class classifier connects the selected units with an appropriate relation label. They evaluate their approach on the RST-DT corpus (Carlson et al., 2002) of news articles. On a different genre of instructional texts, Subba and Di-Eugenio (2009) propose a shift-reduce parser that relies on a classifier for relation labeling. Their classifier uses Inductive Logic Programming (ILP) to learn first-order logic rules from a set of features including *compositional semantics*. In this work, we address the limitations of these models (described in Section 1) introducing our novel discourse parser.

### 3 Our Discourse Parsing Framework

Given a document with sentences already segmented into EDUs, the discourse parsing problem is determining which discourse units (EDUs or larger units) to relate (i.e., the structure), and how to relate them (i.e., the labels or the discourse relations) in the resulting DT. Since we already have an accurate sentence-level discourse parser (Joty et al., 2012), a straightforward approach to document-level parsing could be to simply apply this parser to the whole document. However this strategy would be problematic because of scalability and modeling issues. Note that the number of valid trees grows exponentially with the number of EDUs in a document.<sup>1</sup> Therefore, an exhaustive search over the valid trees is often unfeasible, even for relatively small documents.

For modeling, the problem is two-fold. On the one hand, it appears that rhetorical relations are distributed differently intra-sententially vs. multi-sententially. For example, Figure 2 shows a comparison between the two distributions of six most

<sup>1</sup>For  $n + 1$  EDUs, the number of valid discourse trees is actually the *Catalan number*  $C_n$ .

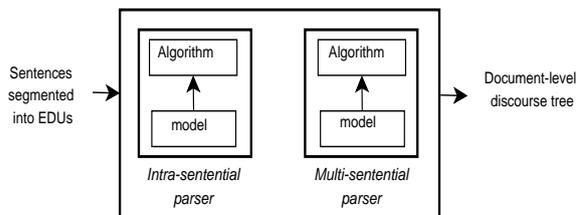


Figure 3: Discourse parsing framework.

frequent relations on a development set containing 20 randomly selected documents from RST-DT. Notice that relations *Attribution* and *Same-Unit* are more frequent than *Joint* in intra-sentential case, whereas *Joint* is more frequent than the other two in multi-sentential case. On the other hand, different kinds of features are applicable and informative for intra-sentential vs. multi-sentential parsing. For example, syntactic features like *dominance sets* (Soricut and Marcu, 2003) are extremely useful for sentence-level parsing, but are not even applicable in multi-sentential case. Likewise, *lexical chain features* (Sporleder and Lascarides, 2004), that are useful for multi-sentential parsing, are not applicable at the sentence level.

Based on these observations, our discourse parsing framework comprises two separate modules: an *intra-sentential parser* and a *multi-sentential parser* (Figure 3). First, the intra-sentential parser produces one or more discourse sub-trees for each sentence. Then, the multi-sentential parser generates a full DT for the document from these sub-trees. Both of our parsers have the same two components: a *parsing model* assigns a probability to every possible DT, and a *parsing algorithm* identifies the most probable DT among the candidate DTs in that scenario. While the two models are rather different, the same parsing algorithm is shared by the two modules. Staging multi-sentential parsing on top of intra-sentential parsing in this way allows us to exploit the strong correlation between the text structure and the DT structure as explained in detail in Section 5. Before describing our parsing models and the parsing algorithm, we introduce some terminology that we will use throughout the paper.

Following (Joty et al., 2012), a DT can be formally represented as a set of constituents of the form  $R[i, m, j]$ , referring to a rhetorical relation  $R$  between the discourse unit containing EDUs  $i$  through  $m$  and the unit containing EDUs  $m+1$  through  $j$ . For example, the DT for the second sentence in Figure 1 can be represented as

$\{Elaboration-NS[4,4,5], Same-Unit-NN[4,5,6]\}$ . Notice that a relation  $R$  also specifies the nuclearity statuses of the discourse units involved, which can be one of *Nucleus-Satellite (NS)*, *Satellite-Nucleus (SN)* and *Nucleus-Nucleus (NN)*.

#### 4 Parsing Models and Parsing Algorithm

The job of our intra-sentential and multi-sentential parsing models is to assign a probability to each of the constituents of all possible DTs at the sentence level and at the document level, respectively. Formally, given the model parameters  $\Theta$ , for each possible constituent  $R[i, m, j]$  in a candidate DT at the sentence or document level, the parsing model estimates  $P(R[i, m, j]|\Theta)$ , which specifies a joint distribution over the label  $R$  and the structure  $[i, m, j]$  of the constituent.

##### 4.1 Intra-Sentential Parsing Model

Recently, we proposed a novel parsing model for sentence-level discourse parsing (Joty et al., 2012), that outperforms previous approaches by effectively modeling sequential dependencies along with structure and labels jointly. Below we briefly describe the parsing model, and show how it is applied to obtain the probabilities of all possible DT constituents at the sentence level.

Figure 4 shows the intra-sentential parsing model expressed as a Dynamic Conditional Random Field (DCRF) (Sutton et al., 2007). The observed nodes  $U_j$  in a sequence represent the discourse units (EDUs or larger units). The first layer of hidden nodes are the structure nodes, where  $S_j \in \{0, 1\}$  denotes whether two adjacent discourse units  $U_{j-1}$  and  $U_j$  should be connected or not. The second layer of hidden nodes are the relation nodes, with  $R_j \in \{1 \dots M\}$  denoting the relation between two adjacent units  $U_{j-1}$  and  $U_j$ , where  $M$  is the total number of relations in the relation set. The connections between adjacent nodes in a hidden layer encode sequential dependencies between the respective hidden nodes, and can enforce constraints such as the fact that a  $S_j = 1$  must not follow a  $S_{j-1} = 1$ . The connections between the two hidden layers model the structure and the relation of a DT (sentence-level) constituent jointly.

To obtain the probability of the constituents of all candidate DTs for a sentence, we apply the parsing model recursively at different levels of the DT and compute the posterior marginals over the relation-structure pairs. To illustrate the

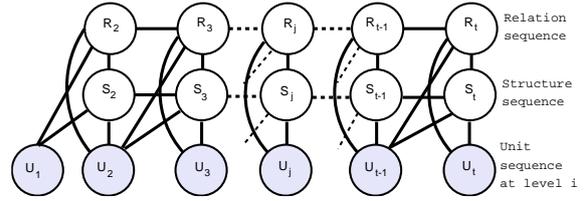


Figure 4: A chain-structured DCRF as our intra-sentential parsing model.

process, let us assume that the sentence contains four EDUs. At the first (bottom) level, when all the units are the EDUs, there is only one possible unit sequence to which we apply our DCRF model (Figure 5(a)). We compute the posterior marginals  $P(R_2, S_2=1|e_1, e_2, e_3, e_4, \Theta)$ ,  $P(R_3, S_3=1|e_1, e_2, e_3, e_4, \Theta)$  and  $P(R_4, S_4=1|e_1, e_2, e_3, e_4, \Theta)$  to obtain the probability of the constituents  $R[1, 1, 2]$ ,  $R[2, 2, 3]$  and  $R[3, 3, 4]$ , respectively. At the second level, there are three possible unit sequences  $(e_{1:2}, e_3, e_4)$ ,  $(e_1, e_{2:3}, e_4)$  and  $(e_1, e_2, e_{3:4})$ . Figure 5(b) shows their corresponding DCRFs. The posterior marginals  $P(R_3, S_3=1|e_{1:2}, e_3, e_4, \Theta)$ ,  $P(R_{2:3}, S_{2:3}=1|e_1, e_{2:3}, e_4, \Theta)$ ,  $P(R_4, S_4=1|e_1, e_{2:3}, e_4, \Theta)$  and  $P(R_{3:4}, S_{3:4}=1|e_1, e_2, e_{3:4}, \Theta)$  computed from the three sequences correspond to the probability of the constituents  $R[1, 2, 3]$ ,  $R[1, 1, 3]$ ,  $R[2, 3, 4]$  and  $R[2, 2, 4]$ , respectively. Similarly, we attain the probability of the constituents  $R[1, 1, 4]$ ,  $R[1, 2, 4]$  and  $R[1, 3, 4]$  by computing their respective posterior marginals from the three possible sequences at the third (top) level.

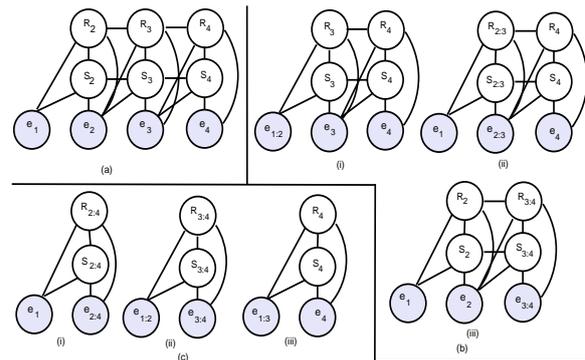


Figure 5: Our parsing model applied to the sequences at different levels of a sentence-level DT. (a) Only possible sequence at the first level, (b) Three possible sequences at the second level, (c) Three possible sequences at the third level.

At this point what is left to be explained is how we generate all possible sequences for a given number of EDUs in a sentence. Algorithm 1 demonstrates how we do that. More specifically, to compute the probabilities of each DT con-

stituent  $R[i, k, j]$ , we need to generate sequences like  $(e_1, \dots, e_{i-1}, e_{i:k}, e_{k+1:j}, e_{j+1}, \dots, e_n)$  for  $1 \leq i \leq k < j \leq n$ . In doing so, we may generate some duplicate sequences. Clearly, the sequence  $(e_1, \dots, e_{i-1}, e_{i:i}, e_{i+1:j}, e_{j+1}, \dots, e_n)$  for  $1 \leq i \leq k < j < n$  is already considered for computing the probability of  $R[i+1, j, j+1]$ . Therefore, it is a duplicate sequence that we exclude from our list of all possible sequences.

```

Input: Sequence of EDUs:  $(e_1, e_2, \dots, e_n)$ 
Output: List of sequences:  $L$ 
for  $i = 1 \rightarrow n - 1$  do
  for  $j = i + 1 \rightarrow n$  do
    if  $j == n$  then
      for  $k = i \rightarrow j - 1$  do
         $L.append$ 
         $((e_1, \dots, e_{i-1}, e_{i:k}, e_{k+1:j}, e_{j+1}, \dots, e_n))$ 
      end
    else
      for  $k = i + 1 \rightarrow j - 1$  do
         $L.append$ 
         $((e_1, \dots, e_{i-1}, e_{i:k}, e_{k+1:j}, e_{j+1}, \dots, e_n))$ 
      end
    end
  end
end

```

**Algorithm 1:** Generating all possible sequences for a sentence with  $n$  EDUs.

Once we obtain the probability of all possible DT constituents, the discourse sub-trees for the sentences are built by applying an optimal probabilistic parsing algorithm (Section 4.4) using one of the methods described in Section 5.

## 4.2 Multi-Sentential Parsing Model

Given the discourse units (sub-trees) for all the sentences of a document, a simple approach to build the rhetorical tree of the document would be to apply a new DCRF model, similar to the one in Figure 4 (with different parameters), to all the possible sequences generated from these units to infer the probability of all possible higher-order constituents. However, the number of possible sequences and their length increase with the number of sentences in a document. For example, assuming that each sentence has a well-formed DT, for a document with  $n$  sentences, Algorithm 1 generates  $O(n^3)$  sequences, where the sequence at the bottom level has  $n$  units, each of the sequences at the second level has  $n-1$  units, and so on. Since the model in Figure 4 has a “fat” chain structure,

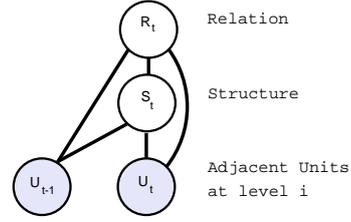


Figure 6: A CRF as a multi-sentential parsing model.

we could use forwards-backwards algorithm for exact inference in this model (Sutton and McCallum, 2012). However, forwards-backwards on a sequence containing  $T$  units costs  $O(TM^2)$  time, where  $M$  is the number of relations in our relation set. This makes the chain-structured DCRF model impractical for multi-sentential parsing of long documents, since learning requires to run inference on every training sequence with an overall time complexity of  $O(TM^2n^3)$  per document.

Our model for multi-sentential parsing is shown in Figure 6. The two observed nodes  $U_{t-1}$  and  $U_t$  are two adjacent discourse units. The (hidden) structure node  $S \in \{0, 1\}$  denotes whether the two units should be connected or not. The hidden node  $R \in \{1 \dots M\}$  represents the relation between the two units. Notice that like the previous model, this is also an undirected graphical model. It becomes a CRF if we directly model the hidden (output) variables by conditioning its clique potential (or factor)  $\phi$  on the observed (input) variables:

$$P(R_t, S_t | \mathbf{x}, \Theta) = \frac{1}{Z(\mathbf{x}, \Theta)} \phi(R_t, S_t | \mathbf{x}, \Theta) \quad (1)$$

where  $\mathbf{x}$  represents input features extracted from the observed variables  $U_{t-1}$  and  $U_t$ , and  $Z(\mathbf{x}, \Theta)$  is the partition function. We use a log-linear representation of the factor:

$$\phi(R_t, S_t | \mathbf{x}, \Theta) = \exp(\Theta^T f(R_t, S_t, \mathbf{x})) \quad (2)$$

where  $f(R_t, S_t, \mathbf{x})$  is a feature vector derived from the input features  $\mathbf{x}$  and the labels  $R_t$  and  $S_t$ , and  $\Theta$  is the corresponding weight vector. Although, this model is similar in spirit to the model in Figure 4, we now break the chain structure, which makes the inference much faster (i.e., complexity of  $O(M^2)$ ). Breaking the chain structure also allows us to balance the data for training (equal number instances with  $S=1$  and  $S=0$ ), which dramatically reduces the learning time of the model.

We apply our model to all possible adjacent units at all levels for the multi-sentential case, and

compute the posterior marginals of the relation-structure pairs  $P(R_t, S_t=1|U_{t-1}, U_t, \Theta)$  to obtain the probability of all possible DT constituents.

### 4.3 Features Used in our Parsing Models

Table 1 summarizes the features used in our parsing models, which are extracted from two adjacent units  $U_{t-1}$  and  $U_t$ . Since most of these features are adopted from previous studies (Joty et al., 2012; Hernault et al., 2010), we briefly describe them.

**Organizational** features include the *length* of the units as the number of EDUs and tokens. It also includes the *distances* of the units from the beginning and end of the sentence (or text in the multi-sentential case). **Text structural** features indirectly capture the correlation between text structure and rhetorical structure by counting the number of *sentence* and *paragraph* boundaries in the units. Discourse markers (e.g., *because*, *although*) carry informative clues for rhetorical relations (Marcu, 2000a). Rather than using a fixed list of discourse markers, we use an empirically learned *lexical N-gram* dictionary following (Joty et al., 2012). This approach has been shown to be more robust and flexible across domains (Biran and Rambow, 2011; Hernault et al., 2010). We also include part-of-speech (*POS*) tags for the beginning and end  $N$  tokens in a unit.

<b>8 Organizational features</b>	<b><i>Intra &amp; Multi-Sentential</i></b>
Number of EDUs in <i>unit 1</i> (or <i>unit 2</i> ).	
Number of tokens in <i>unit 1</i> (or <i>unit 2</i> ).	
Distance of unit 1 in EDUs to the <i>beginning</i> (or to the <i>end</i> ).	
Distance of unit 2 in EDUs to the <i>beginning</i> (or to the <i>end</i> ).	
<b>4 Text structural features</b>	<b><i>Multi-Sentential</i></b>
Number of sentences in <i>unit 1</i> (or <i>unit 2</i> ).	
Number of paragraphs in <i>unit 1</i> (or <i>unit 2</i> ).	
<b>8 N-gram features <math>N \in \{1, 2, 3\}</math></b>	<b><i>Intra &amp; Multi-Sentential</i></b>
<i>Beginning</i> (or <i>end</i> ) lexical N-grams in unit 1.	
<i>Beginning</i> (or <i>end</i> ) lexical N-grams in unit 2.	
<i>Beginning</i> (or <i>end</i> ) POS N-grams in unit 1.	
<i>Beginning</i> (or <i>end</i> ) POS N-grams in unit 2.	
<b>5 Dominance set features</b>	<b><i>Intra-Sentential</i></b>
Syntactic labels of the <i>head</i> node and the <i>attachment</i> node.	
Lexical heads of the <i>head</i> node and the <i>attachment</i> node.	
<i>Dominance relationship</i> between the two units.	
<b>8 Lexical chain features</b>	<b><i>Multi-Sentential</i></b>
Number of chains start in unit 1 and end in unit 2.	
Number of chains <i>start</i> (or <i>end</i> ) in <i>unit 1</i> (or in <i>unit 2</i> ).	
Number of chains skipping both unit 1 and unit 2.	
Number of chains skipping <i>unit 1</i> (or <i>unit 2</i> ).	
<b>2 Contextual features</b>	<b><i>Intra &amp; Multi-Sentential</i></b>
<i>Previous</i> and <i>next</i> feature vectors.	
<b>2 Substructure features</b>	<b><i>Intra &amp; Multi-Sentential</i></b>
Root nodes of the <i>left</i> and <i>right</i> rhetorical sub-trees.	

Table 1: Features used in our parsing models.

Lexico-syntactic features **dominance sets** (Soricut and Marcu, 2003) are very effective for intra-sentential parsing. We include *syntactic labels* and *lexical heads* of head and attachment nodes along with their *dominance relationship* as features. **Lexical chains** (Morris and Hirst, 1991) are sequences of semantically related words that can indicate topic shifts. Features extracted from lexical chains have been shown to be useful for finding paragraph-level discourse structure (Sporleder and Lascarides, 2004). We compute lexical chains for a document following the approach proposed in (Galley and McKeown, 2003), that extracts lexical chains after performing word sense disambiguation. Following (Joty et al., 2012), we also encode *contextual* and *rhetorical sub-structure* features in our models. The rhetorical sub-structure features incorporate hierarchical dependencies between DT constituents.

### 4.4 Parsing Algorithm

Given the probability of all possible DT constituents in the intra-sentential and multi-sentential scenarios, the job of the parsing algorithm is to find the most probable DT for that scenario. Following (Joty et al., 2012), we implement a probabilistic CKY-like bottom-up algorithm for computing the most likely parse using dynamic programming. Specifically, with  $n$  discourse units, we use the upper-triangular portion of the  $n \times n$  dynamic programming table  $D$ . Given  $U_x(0)$  and  $U_x(1)$  are the start and end EDU Ids of unit  $U_x$ :

$$D[i, j] = P(R[U_i(0), U_k(1), U_j(1)]) \quad (3)$$

where,  $k = \operatorname{argmax}_{i \leq p \leq j} P(R[U_i(0), U_p(1), U_j(1)])$ .

Note that, in contrast to previous studies on document-level parsing (Hernault et al., 2010; Subba and Di-Eugenio, 2009; Marcu, 2000b), which use a greedy algorithm, our approach finds a discourse tree that is globally optimal.

## 5 Document-level Parsing Approaches

Now that we have presented our intra-sentential and our multi-sentential parsers, we are ready to describe how they can be effectively combined to perform document-level discourse analysis. Recall that a key motivation for a two-stage parsing is that it allows us to capture the correlation between text structure and discourse structure in a scalable, modular and flexible way. Below we describe two different approaches to model this correlation.

## 5.1 1S-1S (1 Sentence-1 Sub-tree)

A key finding from several previous studies on sentence-level discourse analysis is that most sentences have a well-formed discourse sub-tree in the full document-level DT (Joty et al., 2012; Fisher and Roark, 2007). For example, Figure 7(a) shows 10 EDUs in 3 sentences (see boxes), where the DTs for the sentences obey their respective sentence boundaries. The 1S-1S approach aims to maximally exploit this finding. It first constructs a DT for every sentence using our intra-sentential parser, and then it provides our multi-sentential parser with the sentence-level DTs to build the rhetorical parse for the whole document.

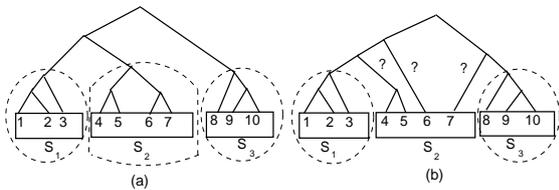


Figure 7: Two possible DTs for three sentences.

## 5.2 Sliding Window

While the assumption made by 1S-1S clearly simplifies the parsing process, it totally ignores the cases where discourse structures violate sentence boundaries. For example, in the DT shown in Figure 7(b), sentence  $S_2$  does not have a well-formed sub-tree because some of its units attach to the left (4-5, 6) and some to the right (7). Vliet and Redeker (2011) call these cases as ‘leaky’ boundaries. Even though less than 5% of the sentences have leaky boundaries in RST-DT, in other corpora this can be true for a larger portion of the sentences. For example, we observe over 12% sentences with leaky boundaries in the Instructional corpus of (Subba and Di-Eugenio, 2009). However, we notice that in most cases where discourse structures violate sentence boundaries, its units are merged with the units of its adjacent sentences, as in Figure 7(b). For example, this is true for 75% cases in our development set containing 20 news articles from RST-DT and for 79% cases in our development set containing 20 how-to-do manuals from the Instructional corpus. Based on this observation, we propose a sliding window approach.

In this approach, our intra-sentential parser works with a window of two consecutive sentences, and builds a DT for the two sentences. For example, given the three sentences in Figure 7, our

intra-sentential parser constructs a DT for  $S_1$ - $S_2$  and a DT for  $S_2$ - $S_3$ . In this process, each sentence in a document except the first and the last will be associated with two DTs: one with the previous sentence (say  $DT_p$ ) and one with the next (say  $DT_n$ ). In other words, for each non-boundary sentence, we will have two decisions: one from  $DT_p$  and one from  $DT_n$ . Our parser consolidates the two decisions and generates one or more sub-trees for each sentence by checking the following three mutually exclusive conditions one after another:

- *Same in both*: If the sentence has the same (in terms of both structure and labels) well-formed sub-tree in both  $DT_p$  and  $DT_n$ , we take this sub-tree for the sentence. For example, in Figure 8(a),  $S_2$  has the same sub-tree in the two DTs, i.e. a DT for  $S_1$ - $S_2$  and a DT for  $S_2$ - $S_3$ . The two decisions agree on the DT for the sentence.
- *Different but no cross*: If the sentence has a well-formed sub-tree in both  $DT_p$  and  $DT_n$ , but the two sub-trees vary either in structure or in labels, we pick the most probable one. For example, consider the DT for  $S_1$ - $S_2$  in Figure 8(a) and the DT for  $S_2$ - $S_3$  in Figure 8(b). In both cases  $S_2$  has a well-formed sub-tree, but they differ in structure. We pick the sub-tree which has the higher probability in the two dynamic programming tables.

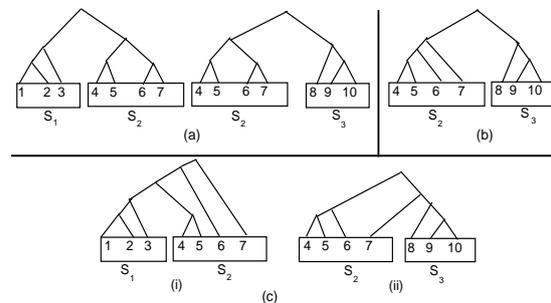


Figure 8: Extracting sub-trees for  $S_2$ .

- *Cross*: If either or both of  $DT_p$  and  $DT_n$  segment the sentence into multiple sub-trees, we pick the one with more sub-trees. For example, consider the two DTs in Figure 8(c). In the DT for  $S_1$ - $S_2$ ,  $S_2$  has three sub-trees (4-5,6,7), whereas in the DT for  $S_2$ - $S_3$ , it has two (4-6,7). So, we extract the three sub-trees for  $S_2$  from the first DT. If the sentence has the same number of sub-trees in both  $DT_p$  and  $DT_n$ , we pick the one with higher probability in the dynamic programming tables.

At the end, the multi-sentential parser takes all these sentence-level sub-trees for a document, and builds a full rhetorical parse for the document.

## 6 Experiments

### 6.1 Corpora

While previous studies on document-level parsing only report their results on a particular corpus, to show the generality of our method, we experiment with texts from two very different genres. Our first corpus is the standard *RST-DT* (Carlson et al., 2002), which consists of 385 Wall Street Journal articles, and is partitioned into a training set of 347 documents and a test set of 38 documents. 53 documents, selected from both sets were annotated by two annotators, based on which we measure human agreement. In *RST-DT*, the original 25 rhetorical relations defined by (Mann and Thompson, 1988) are further divided into a set of 18 coarser relation classes with 78 finer-grained relations. Our second corpus is the *Instructional* corpus prepared by (Subba and Di-Eugenio, 2009), which contains 176 how-to-do manuals on home-repair. The corpus was annotated with 26 informational relations (e.g., *Preparation-Act*, *Act-Goal*).

### 6.2 Experimental Setup

We experiment with our discourse parser on the two datasets using our two different parsing approaches, namely 1S-1S and the sliding window. We compare our approach with *HILDA* (Hernault et al., 2010) on *RST-DT*, and with the *ILP*-based approach of (Subba and Di-Eugenio, 2009) on the *Instructional* corpus, since they are the state-of-the-art on the respective genres. On *RST-DT*, the standard split was used for training and testing purposes. The results for *HILDA* were obtained by running the system with default settings on the same inputs we provided to our system. Since we could not run the *ILP*-based system of (Subba and Di-Eugenio, 2009) (not publicly available) on the *Instructional* corpus, we report the performances presented in their paper. They used 151 documents for training and 25 documents for testing. Since we did not have access to their particular split, we took 5 random samples of 151 documents for training and 25 documents for testing, and report the average performance over the 5 test sets.

To evaluate the parsing performance, we use the standard unlabeled (i.e., hierarchical spans) and labeled (i.e., nuclearity and relation) precision, recall and F-score as described in (Marcu, 2000b). To compare with previous studies, our experiments on *RST-DT* use the 18 coarser relations. After attaching the nuclearity statuses (NS,

SN, NN) to these relations, we get 41 distinct relations. Following (Subba and Di-Eugenio, 2009) on the *Instructional* corpus, we use 26 relations, and treat the reversals of non-commutative relations as separate relations. That is, *Goal-Act* and *Act-Goal* are considered as two different relations. Attaching the nuclearity statuses to these relations gives 76 distinct relations. Analogous to previous studies, we map the *n*-ary relations (e.g., *Joint*) into nested right-branching binary relations.

### 6.3 Results and Error Analysis

Table 2 presents F-score parsing results for our parsers and the existing systems on the two corpora.<sup>2</sup> On both corpora, our parser, namely, 1S-1S (TSP 1-1) and sliding window (TSP SW), outperform existing systems by a wide margin ( $p < 7.1e-05$ ).<sup>3</sup> On *RST-DT*, our parsers achieve absolute F-score improvements of 8%, 9.4% and 11.4% in span, nuclearity and relation, respectively, over *HILDA*. This represents relative error reductions of 32%, 23% and 21% in span, nuclearity and relation, respectively. Our results are also close to the upper bound, i.e. human agreement on this corpus.

On the *Instructional* genre, our parsers deliver absolute F-score improvements of 10.5%, 13.6% and 8.14% in span, nuclearity and relations, respectively, over the *ILP*-based approach. Our parsers, therefore, reduce errors by 36%, 27% and 13% in span, nuclearity and relations, respectively.

If we compare the performance of our parsers on the two corpora, we observe higher results on *RST-DT*. This can be explained in at least two ways. First, the *Instructional* corpus has a smaller amount of data with a larger set of relations (76 when nuclearity attached). Second, some frequent relations are (semantically) very similar (e.g., *Preparation-Act*, *Step1-Step2*), which makes it difficult even for the human annotators to distinguish them (Subba and Di-Eugenio, 2009).

Comparison between our two models reveals that TSP SW significantly outperforms TSP 1-1 only in finding the right structure on both corpora ( $p < 0.01$ ). Not surprisingly, the improvement is higher on the *Instructional* corpus. A likely explanation is that the *Instructional* corpus contains more leaky boundaries (12%), allowing the sliding

<sup>2</sup>Precision, Recall and F-score are the same when manual segmentation is used (see Marcu, (2000b), page 143).

<sup>3</sup>Since we did not have access to the output or to the system of (Subba and Di-Eugenio, 2009), we were not able to perform a significance test on the *Instructional* corpus.

Metrics	RST-DT				Instructional		
	HILDA	TSP 1-1	TSP SW	Human	ILP	TSP 1-1	TSP SW
Span	74.68	<b>82.47*</b>	<b>82.74*†</b>	88.70	70.35	<b>79.67</b>	<b>80.88†</b>
Nuclearity	58.99	<b>68.43*</b>	<b>68.40*</b>	77.72	49.47	<b>63.03</b>	<b>63.10</b>
Relation	44.32	<b>55.73*</b>	<b>55.71*</b>	65.75	35.44	<b>43.52</b>	<b>43.58</b>

Table 2: Parsing results of different models using manual (gold) segmentation. Performances significantly superior to HILDA (with  $p < 7.1e-05$ ) are denoted by \*. Significant differences between TSP 1-1 and TSP SW (with  $p < 0.01$ ) are denoted by †.

	T-C	T-O	T-CM	M-M	CMP	EV	SU	CND	EN	CA	TE	EX	BA	CO	JO	S-U	AT	EL
T-C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
T-O	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T-CM	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	7
M-M	0	0	0	10	0	0	0	0	0	0	0	1	1	0	0	0	1	3
CMP	0	0	0	1	4	0	0	1	0	1	0	3	3	0	1	1	0	2
EV	0	0	0	0	0	0	0	0	0	0	0	2	0	0	2	0	2	11
SU	0	0	0	0	0	0	8	0	0	0	0	0	0	0	1	0	0	12
CND	0	0	0	0	0	0	0	22	0	0	0	0	1	3	0	0	3	2
EN	0	0	0	0	0	0	1	24	1	0	0	0	0	0	0	0	1	7
CA	0	0	0	0	0	0	0	2	3	0	4	2	2	7	0	3	11	
TE	0	0	0	1	0	0	1	2	0	7	1	9	1	9	0	3	4	
EX	0	0	0	1	0	0	0	1	5	0	12	0	1	3	0	3	12	
BA	0	0	0	1	0	0	0	1	0	1	4	1	19	2	6	1	5	12
CO	0	0	0	1	2	0	0	2	0	1	3	2	2	33	7	0	0	9
JO	0	0	0	0	0	0	1	2	0	1	1	1	1	2	57	1	0	13
S-U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	85	1	0
AT	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	3	272	9
EL	0	1	0	0	0	0	0	0	14	6	1	8	1	0	8	2	2	359

Figure 9: Confusion matrix for relation labels on the RST-DT test set. Y-axis represents *true* and X-axis represents *predicted* relations. The relations are Topic-Change (T-C), Topic-Comment (T-CM), Textual Organization (T-O), Manner-Means (M-M), Comparison (CMP), Evaluation (EV), Summary (SU), Condition (CND), Enablement (EN), Cause (CA), Temporal (TE), Explanation (EX), Background (BA), Contrast (CO), Joint (JO), Same-Unit (S-U), Attribution (AT) and Elaboration (EL).

window approach to be more effective in finding those, without inducing much noise for the labels. This clearly demonstrates the potential of TSP SW for datasets with even more leaky boundaries e.g., the Dutch (Vliet and Redeker, 2011) and the German Potsdam (Stede, 2004) corpora.

Error analysis reveals that although TSP SW finds more correct structures, a corresponding improvement in labeling relations is not present because in a few cases, it tends to induce noise from the neighboring sentences for the labels. For example, when parsing was performed on the first sentence in Figure 1 in isolation using 1S-1S, our parser rightly identifies the *Contrast* relation between EDUs 2 and 3. But, when it is considered with its neighboring sentences by the sliding window, the parser labels it as *Elaboration*. A promising strategy to deal with this and similar problems that we plan to explore in future, is to apply both approaches to each sentence and combine them by consolidating three probabilistic decisions, i.e. the one from 1S-1S and the two from sliding window.

To further analyze the errors made by our parser on the hardest task of relation labeling, Figure 9 presents the confusion matrix for TSP 1-1 on the RST-DT test set. The relation labels are ordered according to their frequency in the RST-DT training set. In general, the errors are produced by two different causes acting together: (i) imbalanced distribution of the relations, and (ii) semantic similarity between the relations. The most frequent relation *Elaboration* tends to mislead others especially, the ones which are semantically similar (e.g., *Explanation*, *Background*) and less frequent (e.g., *Summary*, *Evaluation*). The relations which are semantically similar mislead each other (e.g., *Temporal:Background*, *Cause:Explanation*).

These observations suggest two ways to improve our parser. We would like to employ a more robust method (e.g., *ensemble* methods with *bagging*) to deal with the imbalanced distribution of relations, along with taking advantage of a richer semantic knowledge (e.g., compositional semantics) to cope with the errors caused by semantic similarity between the rhetorical relations.

## 7 Conclusion

In this paper, we have presented a novel discourse parser that applies an optimal parsing algorithm to probabilities inferred from two CRF models: one for intra-sentential parsing and the other for multi-sentential parsing. The two models exploit their own informative feature sets and the distributional variations of the relations in the two parsing conditions. We have also presented two novel approaches to combine them effectively. Empirical evaluations on two different genres demonstrate that our approach yields substantial improvement over existing methods in discourse parsing.

## Acknowledgments

We are grateful to Frank Tompa and the anonymous reviewers for their comments, and the NSERC BIN and CGS-D for financial support.

## References

- O. Biran and O. Rambow. 2011. Identifying Justifications in Written Dialogs by Classifying Text as Argumentative. *International Journal of Semantic Computing*, 5(4):363–381.
- L. Carlson, D. Marcu, and M. Okurowski. 2002. RST Discourse Treebank (RST-DT) LDC2002T07. *Linguistic Data Consortium, Philadelphia*.
- V. Feng and G. Hirst. 2012. Text-level Discourse Parsing with Rich Linguistic Features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, ACL '12, pages 60–68, Jeju Island, Korea. Association for Computational Linguistics.
- S. Fisher and B. Roark. 2007. The Utility of Parse-derived Features for Automatic Discourse Segmentation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, ACL '07, pages 488–495, Prague, Czech Republic. Association for Computational Linguistics.
- M. Galley and K. McKeown. 2003. Improving Word Sense Disambiguation in Lexical Chaining. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI '07, pages 1486–1488, Acapulco, Mexico.
- H. Hernault, H. Prendinger, D. duVerle, and M. Ishizuka. 2010. HILDA: A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 1(3):1–33.
- S. Joty, G. Carenini, and R. T. Ng. 2012. A Novel Discriminative Framework for Sentence-Level Discourse Analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 904–915, Jeju Island, Korea. Association for Computational Linguistics.
- H. LeThanh, G. Abeysinghe, and C. Huyck. 2004. Generating Discourse Structures for Written Texts. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Geneva, Switzerland. Association for Computational Linguistics.
- A. Louis, A. Joshi, and A. Nenkova. 2010. Discourse Indicators for Content Selection in Summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '10, pages 147–156, Tokyo, Japan. Association for Computational Linguistics.
- W. Mann and S. Thompson. 1988. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3):243–281.
- D. Marcu and A. Echiabi. 2002. An Unsupervised Approach to Recognizing Discourse Relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 368–375. Association for Computational Linguistics.
- D. Marcu. 2000a. The Rhetorical Parsing of Unrestricted Texts: A Surface-based Approach. *Computational Linguistics*, 26:395–448.
- D. Marcu. 2000b. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, MA, USA.
- J. Morris and G. Hirst. 1991. Lexical Cohesion Computed by Thesaural Relations as an Indicator of Structure of Text. *Computational Linguistics*, 17(1):21–48.
- R. Prasad, A. Joshi, N. Dinesh, A. Lee, E. Miltsakaki, and B. Webber. 2005. The Penn Discourse Treebank as a Resource for Natural Language Generation. In *Proceedings of the Corpus Linguistics Workshop on Using Corpora for Natural Language Generation*, pages 25–32, Birmingham, U.K.
- S. Somasundaran, 2010. *Discourse-Level Relations for Opinion Analysis*. PhD thesis, University of Pittsburgh.
- R. Soricut and D. Marcu. 2003. Sentence Level Discourse Parsing Using Syntactic and Lexical Information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL-HLT '03, pages 149–156, Edmonton, Canada. Association for Computational Linguistics.
- C. Sporleder and M. Lapata. 2005. Discourse Chunking and its Application to Sentence Compression. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 257–264, Vancouver, British Columbia, Canada. Association for Computational Linguistics.
- C. Sporleder and A. Lascarides. 2004. Combining Hierarchical Clustering and Machine Learning to Predict High-Level Discourse Structure. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Geneva, Switzerland. Association for Computational Linguistics.
- M. Stede. 2004. The Potsdam Commentary Corpus. In *Proceedings of the ACL-04 Workshop on Discourse Annotation*, Barcelona. Association for Computational Linguistics.
- R. Subba and B. Di-Eugenio. 2009. An Effective Discourse Parser that Uses Rich Linguistic Information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL '09, pages 566–574, Boulder, Colorado. Association for Computational Linguistics.

- C. Sutton and A. McCallum. 2012. An Introduction to Conditional Random Fields. *Foundations and Trends in Machine Learning*, 4(4):267–373.
- C. Sutton, A. McCallum, and K. Rohanimanesh. 2007. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. *Journal of Machine Learning Research (JMLR)*, 8:693–723.
- S. Verberne, L. Boves, N. Oostdijk, and P. Coppen. 2007. Evaluating Discourse-based Answer Extraction for Why-question Answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 735–736, Amsterdam, The Netherlands. ACM.
- N. Vliet and G. Redeker. 2011. Complex Sentences as Leaky Units in Discourse Parsing. In *Proceedings of Constraints in Discourse*, Agay-Saint Raphael, September.

# Improving pairwise coreference models through feature space hierarchy learning

**Emmanuel Lassalle**

Alpage Project-team

INRIA & Univ. Paris Diderot

Sorbonne Paris Cité, F-75205 Paris

emmanuel.lassalle@ens-lyon.org

**Pascal Denis**

Magnet Project

INRIA Lille - Nord Europe

Avenue Heloise, 59650 Villeneuve d'Ascq

pascal.denis@inria.fr

## Abstract

This paper proposes a new method for significantly improving the performance of pairwise coreference models. Given a set of indicators, our method learns how to best separate types of mention pairs into equivalence classes for which we construct distinct classification models. In effect, our approach finds an optimal feature space (derived from a base feature set and indicator set) for discriminating coreferential mention pairs. Although our approach explores a very large space of possible feature spaces, it remains tractable by exploiting the structure of the hierarchies built from the indicators. Our experiments on the *CoNLL-2012 Shared Task* English datasets (gold mentions) indicate that our method is robust relative to different clustering strategies and evaluation metrics, showing large and consistent improvements over a single pairwise model using the same base features. Our best system obtains a competitive 67.2 of average F1 over MUC, B<sup>3</sup>, and CEAF which, despite its simplicity, places it above the mean score of other systems on these datasets.

## 1 Introduction

Coreference resolution is the problem of partitioning a sequence of noun phrases (or *mentions*), as they occur in a natural language text, into a set of referential *entities*. A common approach to this problem is to separate it into two modules: on the one hand, one defines a model for evaluating coreference links, in general a discriminative classifier that detects coreferential mention pairs. On

the other hand, one designs a method for grouping the detected links into a coherent global output (i.e. a partition over the set of entity mentions). This second step is typically achieved using greedy heuristics (McCarthy and Lehnert, 1995; Soon et al., 2001; Ng and Cardie, 2002; Bengston and Roth, 2008), although more sophisticated clustering approaches have been used, too, such as cutting graph methods (Nicolae and Nicolae, 2006; Cai and Strube, 2010) and Integer Linear Programming (ILP) formulations (Klenner, 2007; Denis and Baldridge, 2009). Despite its simplicity, this two-step strategy remains competitive even when compared to more complex models utilizing a global loss (Bengston and Roth, 2008).

In this kind of architecture, the performance of the entire coreference system strongly depends on the quality of the local pairwise classifier.<sup>1</sup> Consequently, a lot of research effort on coreference resolution has focused on trying to boost the performance of the pairwise classifier. Numerous studies are concerned with feature extraction, typically trying to enrich the classifier with more linguistic knowledge and/or more world knowledge (Ng and Cardie, 2002; Kehler et al., 2004; Ponzetto and Strube, 2006; Bengston and Roth, 2008; Versley et al., 2008; Uryupina et al., 2011). A second line of work explores the use of distinct local models for different types of mentions, specifically for different types of *anaphoric* mentions based on their grammatical categories (such as pronouns, proper names, definite descriptions) (Morton, 2000; Ng, 2005; Denis and Baldridge, 2008).<sup>2</sup> An important justification for such spe-

<sup>1</sup>There are however no theoretical guarantees that improving pair classification will always result in overall improvements if the two modules are optimized independently.

<sup>2</sup>Sometimes, distinct sample selections are also adopted

cialized models is (psycho-)linguistic and comes from theoretical findings based on salience or accessibility (Ariel, 1988). It is worth noting that, from a machine learning point of view, this is related to feature extraction in that both approaches in effect recast the pairwise classification problem in higher dimensional feature spaces.

In this paper, we claim that mention pairs should not be processed by a single classifier, and instead should be handled through specific models. But we are furthermore interested in *learning* how to construct and select such differential models. Our argument is therefore based on statistical considerations, rather than on purely linguistic ones<sup>3</sup>. The main question we raise is, given a set of indicators (such as grammatical types, distance between two mentions, or named entity types), how to best partition the pool of mention pair examples in order to best discriminate coreferential pairs from non coreferential ones. In effect, we want to learn the “best” subspaces for our different models: that is, subspaces that are neither too coarse (i.e., unlikely to separate the data well) nor too specific (i.e., prone to data sparseness and noise). We will see that this is also equivalent to selecting a single large adequate feature space by using the data.

Our approach generalizes earlier approaches in important ways. For one thing, the definition of the different models is no longer restricted to grammatical typing (our model allows for various other types of indicators) or to the sole typing of the anaphoric mention (our models can also be specific to a particular type antecedent or to the two types of the mention pair). More importantly, we propose an original method for learning the best set of models that can be built from a given set of indicators and a training set. These models are organized in a hierarchy, wherein each leaf corresponds to a mutually disjoint subset of mention pair examples and the classifier that can be trained from it. Our models are trained using the *Online Passive-Aggressive* algorithm or *PA* (Crammer et al., 2006), a large margin version of the perceptron. Our method is exact in that it explores the full space of hierarchies (of size at least  $2^{2^n}$ ) definable on an indicator sequence, while remaining scalable by exploiting the particular structure of these

---

during the training of the distinct local models (Ng and Cardie, 2002; Uryupina, 2004).

<sup>3</sup>However it should be underlined that the statistical viewpoint is complementary to the linguistic work.

hierarchies with dynamic programming. This approach also performs well, and it largely outperforms the single model. As will be shown based on a variety of experiments on the *CoNLL-2012 Shared Task* English datasets, these improvements are consistent across different evaluation metrics and for the most part independent of the clustering decoder that was used.

The rest of this paper is organized as follows. Section 2 discusses the underlying statistical hypotheses of the standard pairwise model and defines a simple alternative framework that uses a simple separation of mention pairs based on grammatical types. Next, in section 3, we generalize the method by introducing indicator hierarchies and explain how to learn the best models associated with them. Section 4 provides a brief system description and Section 5 evaluates the various models on CoNLL-2012 English datasets.

## 2 Modeling pairs

Pairwise models basically employ one local classifier to decide whether two mentions are coreferential or not. When using machine learning techniques, this involves certain assumptions about the statistical behavior of mention pairs.

### 2.1 Statistical assumptions

Let us adopt a probabilistic point of view to describe the prototype of pairwise models. Given a document, the number of mentions is fixed and each pair of mentions follows a certain distribution (that we partly observe in a feature space). The basic idea of pairwise models is to consider mention pairs independently from each other (that is why a decoder is necessary to enforce transitivity).

If we use a single classifier to process all pairs, then they are supposed to be identically distributed. We claim that pairs should not be processed by a single classifier because they are not identically distributed (or at least the distribution is too complex for the classifier); rather, we should separate different “types” on pairs and create a specific model for each of them.

Separating different kinds of pairs and handling them with different specific models can lead to more accurate global models. For instance, some coreference resolution systems process different kinds of anaphors separately, which suggests for example that pairs containing an anaphoric pronoun behave differently from pairs with non-

pronominal anaphors. One could rely on a rich set of features to capture complex distributions, but here we actually have a rather limited set of elementary features (see section 4) and, for instance, using products of features must be done carefully to avoid introducing noise in the model. Instead of imposing heuristic product of features, we will show that a clever separation of instances leads to significant improvements of the pairwise model.

## 2.2 Feature spaces

### 2.2.1 Definitions

We first introduce the problem more formally. Every pair of mentions  $m_i$  and  $m_j$  is modeled by a random variable:

$$\begin{aligned} P_{ij} : \Omega &\rightarrow \mathcal{X} \times \mathcal{Y} \\ \omega &\mapsto (x_{ij}(\omega), y_{ij}(\omega)) \end{aligned}$$

where  $\Omega$  classically represents randomness,  $\mathcal{X}$  is the space of objects (“mention pairs”) that is not directly observable and  $y_{ij}(\omega) \in \mathcal{Y} = \{+1, -1\}$  are the labels indicating whether  $m_i$  and  $m_j$  are coreferential or not. To lighten the notations, we will not always write the index  $ij$ . Now we define a mapping:

$$\begin{aligned} \phi_{\mathcal{F}} : \mathcal{X} &\rightarrow \mathcal{F} \\ x &\mapsto \mathbf{x} \end{aligned}$$

that casts pairs into a feature space  $\mathcal{F}$  through which we observe them. For us,  $\mathcal{F}$  is simply a vector space over  $\mathbb{R}$  (in our case many features are Boolean; they are cast into  $\mathbb{R}$  as 0 and 1).

For technical coherence, we assume that  $\phi_{\mathcal{F}_1}(x(\omega))$  and  $\phi_{\mathcal{F}_2}(x(\omega))$  have the same values when projected on the feature space  $\mathcal{F}_1 \cap \mathcal{F}_2$ : it means that common features from two feature spaces have the same values.

From this formal point of view, the task of coreference resolution consists in fixing  $\phi_{\mathcal{F}}$ , observing labeled samples  $\{(\phi_{\mathcal{F}}(x), y)_t\}_{t \in TrainSet}$  and, given partially observed new variables  $\{(\phi_{\mathcal{F}}(x))_t\}_{t \in TestSet}$ , recovering the corresponding values of  $y$ .

### 2.2.2 Formalizing the statistical assumptions

We claimed before that all mention pairs seemed not to be identically distributed since, for example, pronouns do not behave like nominals. We can formulate this more rigorously: since the object space  $\mathcal{X}$  is not directly observable, we do not

know its complexity. In particular, when using a mapping to a too small feature space, the classifier cannot capture the distribution very well: the data is too noisy.

Now if we say that pronominal anaphora do not behave like other anaphora, we distinguish two kinds of pair i.e. we state that the distribution of pairs in  $\mathcal{X}$  is a mixture of two distributions, and we deterministically separate pairs to their specific distribution part. In this way, we may separate positive and negative pairs more easily if we cast each kind of pair into a specific feature space. Let us call these feature spaces  $\mathcal{F}_1$  and  $\mathcal{F}_2$ . We can either create two independent classifiers on  $\mathcal{F}_1$  and  $\mathcal{F}_2$  to process each kind of pair or define a single model on a larger feature space  $\mathcal{F} = \mathcal{F}_1 \oplus \mathcal{F}_2$ . If the model is linear (which is our case), these approaches happen to be equivalent.

So we can actually assume that the random variables  $P_{ij}$  are identically distributed, but drawn from a complex mixture. A new issue arises: we need to find a mapping  $\phi_{\mathcal{F}}$  that renders the best view on the distribution of the data.

From a theoretical viewpoint, the higher the dimension of the feature space (imagine taking the direct sum of all feature spaces), the more we get details on the distribution of mention pairs and the more we can expect to separate positives and negatives accurately. In practice, we have to cope with data sparsity: there will not be enough data to properly train a linear model on such a space. Finally, we seek a feature space situated between the two extremes of a space that is too big (sparseness) or too small (noisy data). The core of this work is to define a general method for choosing the most adequate space  $\mathcal{F}$  among a huge number of possibilities when we do not know *a priori* which is the best.

### 2.2.3 Linear models

In this work, we try to linearly separate positive and negative instances in the large space  $\mathcal{F}$  with the *Online Passive-Aggressive* (PA) algorithm (Crammer et al., 2006): the model learns a parameter vector  $\mathbf{w}$  that defines a hyperplane that cuts the space into two parts. The predicted class of a pair  $x$  with feature vector  $\phi_{\mathcal{F}}(x)$  is given by:

$$C_{\mathcal{F}}(x) := \text{sign}(\mathbf{w}^T \cdot \phi_{\mathcal{F}}(x))$$

Linearity implies an equivalence between: (i) separating instances of two types,  $t_1$  and  $t_2$ , in two

independent models with respective feature spaces  $\mathcal{F}_1$  and  $\mathcal{F}_2$  and parameters  $\mathbf{w}^1$  and  $\mathbf{w}^2$ , and (ii) a single model on  $\mathcal{F}_1 \oplus \mathcal{F}_2$ . To see why, let us define the map:

$$\phi_{\mathcal{F}_1 \oplus \mathcal{F}_2}(x) := \begin{cases} \begin{pmatrix} \phi_{\mathcal{F}_1}(x)^T & 0 \end{pmatrix}^T & \text{if } x \text{ typed } t_1 \\ \begin{pmatrix} 0 & \phi_{\mathcal{F}_2}(x)^T \end{pmatrix}^T & \text{if } x \text{ typed } t_2 \end{cases}$$

and the parameter vector  $\mathbf{w} = \begin{pmatrix} \mathbf{w}^1 \\ \mathbf{w}^2 \end{pmatrix} \in \mathcal{F}_1 \oplus \mathcal{F}_2$ . Then we have:

$$C_{\mathcal{F}_1 \oplus \mathcal{F}_2}(x) = \begin{cases} C_{\mathcal{F}_1}(x) & \text{if } x \text{ typed } t_1 \\ C_{\mathcal{F}_2}(x) & \text{if } x \text{ typed } t_2 \end{cases}$$

Now we check that the same property applies when the PA fits its parameter  $\mathbf{w}$ . For each new instance of the training set, the weight is updated according to the following rule<sup>4</sup>:

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w} \in \mathcal{F}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 \text{ s.t. } l(\mathbf{w}; (x_t, y_t)) = 0$$

where  $l(\mathbf{w}; (x_t, y_t)) = \min(0, 1 - y_t(\mathbf{w} \cdot \phi_{\mathcal{F}}(x_t)))$ , so that when  $\mathcal{F} = \mathcal{F}_1 \oplus \mathcal{F}_2$ , the minimum if  $x$  is typed  $t_1$  is  $\mathbf{w}_{t+1} = \begin{pmatrix} \mathbf{w}_{t+1}^1 \\ \mathbf{w}_t^2 \end{pmatrix}$  and if  $x$  is typed  $t_2$  is  $\mathbf{w}_{t+1} = \begin{pmatrix} \mathbf{w}_t^1 \\ \mathbf{w}_{t+1}^2 \end{pmatrix}$  where the  $\mathbf{w}_{t+1}^i$  correspond to the updates in space  $\mathcal{F}_i$  independently from the rest. This result can be extended easily to the case of  $n$  feature spaces. Thus, with a deterministic separation of the data, a large model can be learned using smaller independent models.

### 2.3 An example: separation by *gramtype*

To motivate our approach, we first introduce a simple separation of mention pairs which creates 9 models obtained by considering all possible pairs of grammatical types  $\{\textit{nominal}, \textit{name}, \textit{pronoun}\}$  for *both* mentions in the pair (a similar fine-grained separation can be found in (Chen et al., 2011)). This is equivalent to using 9 different feature spaces  $\mathcal{F}_1, \dots, \mathcal{F}_9$  to capture the global distribution of pairs. With the PA, this is also a single model with feature space  $\mathcal{F} = \mathcal{F}_1 \oplus \dots \oplus \mathcal{F}_9$ . We will call it the GRAMTYPE model.

As we will see in Section 5, these separated models significantly outperform a single model

<sup>4</sup>The parameter is updated to obtain a margin of a least 1. It does not change if the instance is already correctly classified with such margin.

that uses the same base feature set. But we would like to define a method that adapts a feature space to the data by choosing the most adequate separation of pairs.

## 3 Hierarchizing feature spaces

In this section, we have to keep in mind that separating the pairs in different models is the same as building a large feature space in which the parameter  $\mathbf{w}$  can be learned by parts in independent subspaces.

### 3.1 Indicators on pairs

For establishing a structure on feature spaces, we use *indicators* which are deterministic functions on mention pairs with a small number of outputs. Indicators classify pairs in predefined categories in one-to-one correspondence with independent feature spaces. We can reuse some features of the system as indicators, e.g. the grammatical or named entity types. We can also employ functions that are not used as features, e.g. the approximate position of one of the mentions in the text.

The small number of outputs of an indicator is required for practical reasons: if a category of pairs is too refined, the associated feature space will suffer from data sparsity. Accordingly, distance-based indicators must be approximated by coarse histograms. In our experiments the outputs never exceeded a dozen values. One way to reduce the output span of an indicator is to binarize it like binarizing a tree (many possible binarizations). This operation produces a hierarchy of indicators which is exactly the structure we exploit in what follows.

### 3.2 Hierarchies for separating pairs

We define hierarchies as combinations of indicators creating finer categories of mention pairs: given a finite sequence of indicators, a mention pair is classified by applying the indicators successively, each time refining a category into sub-categories, just like in a decision tree (each node having the same number of children as the number of outputs of its indicator). We allow the classification to stop before applying the last indicator, but the behavior must be the same for all the instances. So a hierarchy is basically a sub-tree of the complete decision tree that contains copies of the same indicator at each level.

If all the leaves of the decision tree have the

same depth, this corresponds to taking the Cartesian product of outputs of all indicators for indexing the categories. In that case, we refer to *product-hierarchies*. The GRAMTYPE model can be seen as a two level product-hierarchy (figure 1).

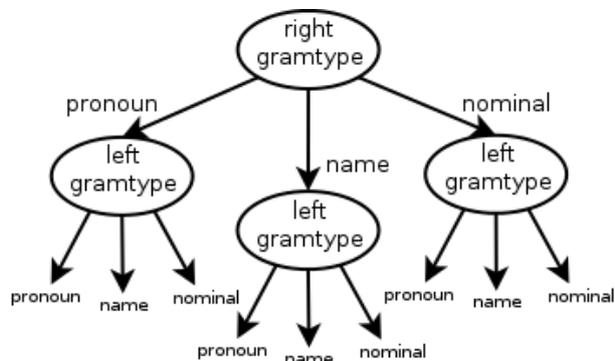


Figure 1: GRAMTYPE seen as a product-hierarchy

Product-hierarchies will be the starting point of our method to find a feature space that fits the data.

Now choosing a relevant sequence of indicators should be achieved through linguistic intuitions and theoretical work (*gramtype* separation is one of them). The system will find by itself the best usage of the indicators when optimizing the hierarchy. The sequence is a parameter of the model.

### 3.3 Relation with feature spaces

Like we did for the GRAMTYPE model, we associate a feature space  $\mathcal{F}_i$  to each leaf of a hierarchy. Likewise, the sum  $\mathcal{F} = \bigoplus_i \mathcal{F}_i$  defines a large feature space. The corresponding parameter  $w$  of the model can be obtained by learning the  $w_i$  in  $\mathcal{F}_i$ .

Given a sequence of indicators, the number of different hierarchies we can define is equal to the number of sub-trees of the complete decision tree (each non-leaf node having all its children). The minimal case is when all indicators are Boolean. The number of full binary trees of height at most  $n$  can be computed by the following recursion:  $T(1) = 1$  and  $T(n + 1) = 1 + T(n)^2$ . So  $T(n) \geq 2^{2^n}$ : even with small values of  $n$ , the number of different hierarchies (or large feature spaces) definable with a sequence of indicators is gigantic (e.g.  $T(10) \approx 3.8 \cdot 10^{90}$ ).

Among all the possibilities for a large feature space, many are irrelevant because for them the data is too sparse or too noisy in some subspaces. We need a general method for finding an adequate space without enumerating and testing each of them.

### 3.4 Optimizing hierarchies

Let us assume now that the sequence of indicators is fixed, and let  $n$  be its length. To find the best feature space among a very high number of possibilities, we need a criterion we can apply without too much additional computation. For that we only evaluate the feature space locally on pairs, i.e. without applying a decoder on the output. We employ 3 measures on pairwise classification results: precision, recall and F1-score. Now selecting the best space for one of these measures can be achieved by using dynamic programming techniques. In the rest of the paper, we will optimize the F1-score.

**Training the hierarchy** Starting from the product-hierarchy, we associate a classifier and its proper feature space to each node of the tree<sup>5</sup>. The classifiers are then trained as follows: for each instance there is a unique path from the root to a leaf of the complete tree. Each classifier situated on the path is updated with this instance. The number of iterations of the Passive-Aggressive is fixed.

**Computing scores** After training, we test all the classifiers on another set of pairs<sup>6</sup>. Again, a classifier is tested on an instance only if it is situated on the path from the root to the leaf associated with the instance. We obtain TP/FP/FN numbers<sup>7</sup> on pair classifications that are sufficient to compute the F1-score. As for training, the data on which a classifier at a given node is evaluated is the same as the union of all data used to evaluate the classifiers corresponding to the children of this node. Thus we are able to compare the scores obtained at a node to the “union of the scores” obtained at its children.

**Cutting down the hierarchy** For the moment we have a complete tree with a classifier at each node. We use a dynamic programming technique to compute the best hierarchy by cutting this tree and only keeping classifiers situated at the leaf. The algorithm assembles the best local models (or feature spaces) together to create larger models. It goes from the leaves to the root and cuts the subtree starting at a node whenever it does not pro-

<sup>5</sup>In the experiments, the classifiers use a copy of a same feature space, but not the same data, which corresponds to crossing the features with the categories of the decision tree.

<sup>6</sup>The training set is cut into two parts, for training and testing the hierarchy. We used 10-fold cross-validation in our experiments.

<sup>7</sup>True positives, false positives and false negatives.

vide a better score than the node itself, or on the contrary propagates the score of the sub-tree when there is an improvement. The details are given in algorithm 1.

```

1 list  $\leftarrow$  list of nodes given by a breadth-first
  search for node in reversed list do
2   if node.children  $\neq \emptyset$  then
3     if sum-score(node.children) >
      node.score then
4       node.TP/FP/FN  $\leftarrow$ 
        sum-num(node.children)
5     else
6       node.children  $\leftarrow \emptyset$ 
7     end
8   end
9 end

```

**Algorithm 1:** Cutting down a hierarchy

Let us briefly discuss the correctness and complexity of the algorithm. Each node is seen two times so the time complexity is linear in the number of nodes which is at least  $\mathcal{O}(2^n)$ . However, only nodes that have encountered at least one training instance are useful and there are  $\mathcal{O}(n \times k)$  such nodes (where  $k$  the size of the training set). So we can optimize the algorithm to run in time  $\mathcal{O}(n \times k)^8$ . If we scan the list obtained by breadth-first search backwards, we are ensured that every node will be processed after its children. (*node.children*) is the set of children of *node*, and (*node.score*) its score. *sum-num* provides TP/FP/FN by simply adding those of the children and *sum-score* computes the score based on these new TP/FP/FN numbers. (line 6) cuts the children of a node when they are not used in the best score. The algorithm thus propagates the best scores from the leaves to the root which finally gives a single score corresponding to the best hierarchy. Only the leaves used to compute the best score are kept, and they define the best hierarchy.

**Relation between cutting and the global feature space** We can see the operation of cutting as replacing a group of subspaces by a single subspace in the sum (see figure 2). So cutting down the product-hierarchy amounts to reducing the global initial feature space in an optimal way.

<sup>8</sup>In our experiments, cutting down the hierarchy was achieved very quickly, and the total training time was about five times longer than with a single model.

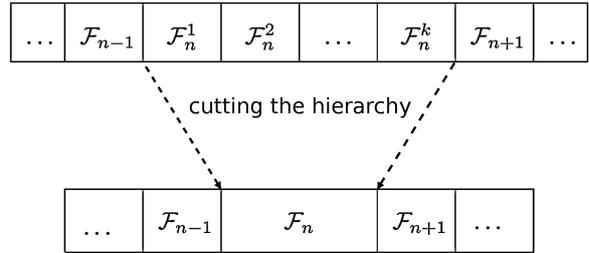


Figure 2: Cutting down the hierarchy reduces the feature space

To sum up, the whole procedure is equivalent to training more than  $\mathcal{O}(2^n)$  perceptrons simultaneously and selecting the best performing.

## 4 System description

Our system consists in the pairwise model obtained by cutting a hierarchy (the PA with selected feature space) and using a greedy decoder to create clusters from the output. It is parametrized by the choice of the initial sequence of indicators.

### 4.1 The base features

We used classical features that can be found in details in (Bengston and Roth, 2008) and (Rahman and Ng, 2011): grammatical type and sub-type of mentions, string match and substring, apposition and copula, distance (number of separating mentions/sentences/words), gender/number match, synonymy/hypernym and animacy (using WordNet), family name (based on lists), named entity types, syntactic features (gold parse) and anaphoricity detection.

### 4.2 Indicators

As indicators we used: left and right grammatical types and subtypes, entity types, a boolean indicating if the mentions are in the same sentence, and a very coarse histogram of distance in terms of sentences. We systematically included right gramtype and left gramtype in the sequences and added other indicators, producing sequences of different lengths. The parameter was optimized by document categories using a development set *after* decoding the output of the pairwise model.

### 4.3 Decoders

We tested 3 classical greedy link selection strategies that form clusters from the classifier decision: Closest-First (merge mentions with their closest coreferent mention on the left) (Soon et al., 2001),

Best-first (merge mentions with the mention on the left having the highest positive score) (Ng and Cardie, 2002; Bengtson and Roth, 2008), and Aggressive-Merge (transitive closure on positive pairs) (McCarthy and Lehnert, 1995). Each of these decoders is typically (although not always) used in tandem with a specific sampling selection at training. Thus, Closest-First for instance is used in combination with a sample selection that generates training instances only for the mentions that occur between the closest antecedent and the anaphor (Soon et al., 2001).

	P	R	F1
SINGLE MODEL	22.28	<b>63.50</b>	32.99
RIGHT-TYPE	29.31	45.23	35.58
GRAMTYPE	39.12	45.83	42.21
BEST HIERARCHY	<b>45.27</b>	51.98	<b>48.40</b>

Table 1: Pairwise scores on CoNLL-2012 test.

## 5 Experiments

### 5.1 Data

We evaluated the system on the English part of the corpus provided in the *CoNLL-2012 Shared Task* (Pradhan et al., 2012), referred to as CoNLL-2012 here. The corpus contains 7 categories of documents (over 2K documents, 1.3M words). We used the official train/dev/test data sets. We evaluated our system in the *closed mode* which requires that only provided data is used.

### 5.2 Settings

Our baselines are a SINGLE MODEL, the GRAM-TYPE model (section 2) and a RIGHT-TYPE model, defined as the first level of the *gramtype* product hierarchy (i.e. grammatical type of the anaphora (Morton, 2000)), with each greedy decoder and also the original sampling with a single model associated with those decoders.

The hierarchies were trained with 10-fold cross-validation on the training set (the hierarchies are cut after cumulating the scores obtained by cross-validation) and their parameters are optimized *by document category* on the development set: the sequence of indicators obtaining the best average score *after* decoding was selected as parameter for the category. The obtained hierarchy is referred to as the BEST HIERARCHY in the results. We fixed the number of iterations for the PA for all models.

In our experiments, we consider only the *gold mentions*. This is a rather idealized setting but our focus is on comparing various pairwise local models rather than on building a full coreference resolution system. Also, we wanted to avoid having to consider too many parameters in our experiments.

### 5.3 Evaluation metrics

We use the three metrics that are most commonly used<sup>9</sup>, namely:

**MUC** (Vilain et al., 1995) computes for each true entity cluster the number of system clusters that are needed to cover it. Precision is this quantity divided by the true cluster size minus one. Recall is obtained by reversing true and predicated clusters. F1 is the harmonic mean.

**B<sup>3</sup>** (Bagga and Baldwin, 1998) computes recall and precision scores for each mention, based on the intersection between the system/true clusters for that mention. Precision is the ratio of the intersection and the true cluster sizes, while recall is the ratio of the intersection to the system cluster sizes. Global recall, precision, and F1 scores are obtained by averaging over the mention scores.

**CEAF** (Luo, 2005) scores are obtained by computing the best one-to-one mapping between the system/true partitions, which is equivalent to finding the best optimal alignment in the bipartite graph formed out of these partitions. We use the  $\phi_4$  similarity function from (Luo, 2005).

These metrics were recently used in the *CoNLL-2011* and *-2012 Shared Tasks*. In addition, these campaigns use an unweighted average over the F1 scores given by the three metrics. Following common practice, we use micro-averaging when reporting our scores for entire datasets.

### 5.4 Results

The results obtained by the system are reported in table 2. The original sampling for the single model associated to Closest-First and Best-First decoder are referred to as SOON and NGCARDIE.

The P/R/F1 pairwise scores before decoding are given in table 1. BEST HIERARCHY obtains a strong improvement in F1 (+15), a better precision and a less significant diminution of recall compared to GRAMTYPE and RIGHT-TYPE.

<sup>9</sup>BLANC metric (Recasens and Hovy, 2011) results are not reported since they are not used to compute the CoNLL-2012 global score. However we can mention that in our experiments, using hierarchies had a positive effect similar to what was observed on B<sup>3</sup> and CEAF.

Closest-First	MUC			B <sup>3</sup>			CEAF			Mean
	P	R	F1	P	R	F1	P	R	F1	
SOON	79.49	<b>93.72</b>	<b>86.02</b>	26.23	<b>89.43</b>	40.56	<b>49.74</b>	19.92	28.44	51.67
SINGLE MODEL	78.95	75.15	77.0	51.88	68.42	59.01	37.79	43.89	40.61	58.87
RIGHT-TYPE	79.36	67.57	72.99	69.43	56.78	62.47	41.17	<b>61.66</b>	49.37	61.61
GRAMTYPE	80.5	71.12	75.52	66.39	61.04	63.6	43.11	59.93	50.15	63.09
BEST HIERARCHY	<b>83.23</b>	73.72	78.19	<b>73.5</b>	67.09	<b>70.15</b>	47.3	60.89	<b>53.24</b>	<b>67.19</b>

Best-First	MUC			B <sup>3</sup>			CEAF			Mean
	P	R	F1	P	R	F1	P	R	F1	
NGCARDIE	<b>81.02</b>	<b>93.82</b>	<b>86.95</b>	23.33	<b>93.92</b>	37.37	<b>40.31</b>	18.97	25.8	50.04
SINGLE MODEL	79.22	73.75	76.39	40.93	75.48	53.08	30.52	37.59	33.69	54.39
RIGHT-TYPE	77.13	65.09	70.60	48.11	66.21	55.73	31.07	47.30	37.50	54.61
GRAMTYPE	77.21	65.89	71.1	49.77	67.19	57.18	32.08	<b>47.83</b>	38.41	55.56
BEST HIERARCHY	78.11	69.82	73.73	<b>53.62</b>	70.86	<b>61.05</b>	35.04	46.67	<b>40.03</b>	<b>58.27</b>

Aggressive-Merge	MUC			B <sup>3</sup>			CEAF			Mean
	P	R	F1	P	R	F1	P	R	F1	
SINGLE MODEL	83.15	<b>88.65</b>	<b>85.81</b>	35.67	<b>88.18</b>	50.79	36.3	28.27	31.78	56.13
RIGHT-TYPE	83.48	89.79	86.52	36.82	88.08	51.93	45.30	33.84	38.74	59.07
GRAMTYPE	83.12	84.27	83.69	44.73	81.58	57.78	45.02	42.94	43.95	61.81
BEST HIERARCHY	<b>83.26</b>	85.2	84.22	<b>45.65</b>	82.48	<b>58.77</b>	<b>46.28</b>	<b>43.13</b>	<b>44.65</b>	<b>62.55</b>

Table 2: CoNLL-2012 test (gold mentions): Closest-First, Best-First and Aggressive-Merge decoders.

Despite the use of greedy decoders, we observe a large positive effect of pair separation in the pairwise models on the outputs. On the mean score, the use of distinct models versus a single model yields F1 increases from 6.4 up to 8.3 depending on the decoder. Irrespective of the decoder being used, GRAMTYPE always outperforms RIGHT-TYPE and single model and is always outperformed by BEST HIERARCHY model.

Interestingly, we see that the increment in pairwise and global score are not proportional: for instance, the strong improvement of F1 between RIGHT-TYPE and GRAMTYPE results in a small amelioration of the global score.

Depending on the document category, we found some variations as to which hierarchy was learned in each setting, but we noticed that parameters starting with right and left gramtypes often produced quite good hierarchies: for instance right gramtype  $\rightarrow$  left gramtype  $\rightarrow$  same sentence  $\rightarrow$  right named entity type.

We observed that product-hierarchies did not performed well without cutting (especially when using longer sequences of indicators, because of data sparsity) and could obtain scores lower than the single model. Hopefully, after cutting them the

results always became better as the resulting hierarchy was more balanced.

Looking at the different metrics, we notice that overall, pair separation improves B<sup>3</sup> and CEAF (but not always MUC) after decoding the output: GRAMTYPE provides a better mean score than the single model, and BEST HIERARCHY gives the highest B<sup>3</sup>, CEAF and mean score.

The best classifier-decoder combination reaches a score of 67.19, which would place it above the mean score (66.41) of the systems that took part in the *CoNLL-2012 Shared Task* (gold mentions track). Except for the first at 77.22, the best performing systems have a score around 68-69. Considering the simple decoding strategy we employed, our current system sets up a strong baseline.

## 6 Conclusion and perspectives

In this paper, we described a method for selecting a feature space among a very large number of choices by using linearity and by combining indicators to separate the instances. We employed dynamic programming on hierarchies of indicators to compute the feature space providing the best pairwise classifications efficiently. We applied this

method to optimize the pairwise model of a coreference resolution system. Using different kinds of greedy decoders, we showed a significant improvement of the system.

Our approach is flexible in that we can use a variety of indicators. In the future we will apply the hierarchies on finer feature spaces to make more accurate optimizations. Observing that the general method of cutting down hierarchies is not restricted to modeling mention pairs, but can be applied to problems having Boolean aspects, we aim at employing hierarchies to address other tasks in computational linguistics (e.g. anaphoricity detection or discourse and temporal relation classification wherein position information may help separating the data).

In this work, we have only considered standard, heuristic linking strategies like Closest-First. So, a natural extension of this work is to combine our method for learning pairwise models with more sophisticated decoding strategies (like *Bestcut* or using ILP). Then we can test the impact of hierarchies with more realistic settings.

Finally, the method for cutting hierarchies should be compared to more general but similar methods, for instance polynomial kernels for SVM and tree-based methods (Hastie et al., 2001). We also plan to extend our method by breaking the symmetry of our hierarchies. Instead of cutting product-hierarchies, we will employ usual techniques to build decision trees<sup>10</sup> and apply our cutting method on their structure. The objective is twofold: first, we will get rid of the sequence of indicators as parameter. Second, we will avoid fragmentation or overfitting (which can arise with classification trees) by deriving an optimal large margin linear model from the tree structure.

## Acknowledgments

We thank the *ACL 2013* anonymous reviewers for their valuable comments.

## References

- M. Ariel. 1988. Referring and accessibility. *Journal of Linguistics*, pages 65–87.
- A. Bagga and B. Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of LREC 1998*, pages 563–566.

<sup>10</sup>(Bansal and Klein, 2012) show good performances of decision trees on coreference resolution.

- Mohit Bansal and Dan Klein. 2012. Coreference semantics from web features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 389–398. Association for Computational Linguistics.
- Eric Bengston and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of EMNLP 2008*, pages 294–303, Honolulu, Hawaii.
- Jie Cai and Michael Strube. 2010. End-to-end coreference resolution via hypergraph partitioning. In *COLING*, pages 143–151.
- Bin Chen, Jian Su, Sinno Jialin Pan, and Chew Lim Tan. 2011. A unified event coreference resolution by integrating multiple resolvers. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 102–110, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.
- Pascal Denis and Jason Baldridge. 2008. Specialized models and ranking for coreference resolution. In *Proceedings of EMNLP 2008*, pages 660–669, Honolulu, Hawaii.
- Pascal Denis and Jason Baldridge. 2009. Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, 43.
- Trevor Hastie, Robert Tibshirani, and J. H. Friedman. 2001. *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. New York: Springer-Verlag.
- A. Kehler, D. Appelt, L. Taylor, and A. Simma. 2004. The (non)utility of predicate-argument frequencies for pronoun interpretation. In *Proceedings of HLT-NAACL 2004*.
- M. Klenner. 2007. Enforcing coherence on coreference sets. In *Proceedings of RANLP 2007*.
- X. Luo. 2005. On coreference resolution performance metrics. In *Proceedings of HLT-NAACL 2005*, pages 25–32.
- J. F. McCarthy and W. G. Lehnert. 1995. Using decision trees for coreference resolution. In *IJCAI*, pages 1050–1055.
- T. Morton. 2000. Coreference for NLP applications. In *Proceedings of ACL 2000*, Hong Kong.
- V. Ng and C. Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of ACL 2002*, pages 104–111.

- V. Ng. 2005. Supervised ranking for pronoun resolution: Some recent improvements. In *Proceedings of AAAI 2005*.
- Cristina Nicolae and Gabriel Nicolae. 2006. Best-cut: A graph algorithm for coreference resolution. In *EMNLP*, pages 275–283.
- Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution. In *Proceedings of the HLT 2006*, pages 192–199, New York City, N.Y.
- Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea, July. Association for Computational Linguistics.
- Ataf Rahman and Vincent Ng. 2011. Narrowing the modeling gap: a cluster-ranking approach to coreference resolution. *J. Artif. Int. Res.*, 40(1):469–521.
- Recasens and Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17:485–510, 9.
- W. M. Soon, H. T. Ng, and D. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.
- Olga Uryupina, Massimo Poesio, Claudio Giuliano, and Kateryna Tymoshenko. 2011. Disambiguation and filtering methods in using web knowledge for coreference resolution. In *FLAIRS Conference*.
- O. Uryupina. 2004. Linguistically motivated sample selection for coreference resolution. In *Proceedings of DAARC 2004*, Furnas.
- Yannick Versley, Alessandro Moschitti, Massimo Poesio, and Xiaofeng Yang. 2008. Coreference systems based on kernels methods. In *COLING*, pages 961–968.
- M. Vilain, J. Burger, J. Aberdeen, D. Connolly, and L. Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings fo the 6th Message Understanding Conference (MUC-6)*, pages 45–52, San Mateo, CA. Morgan Kaufmann.

# Feature-Based Selection of Dependency Paths in Ad Hoc Information Retrieval

**K. Tamsin Maxwell**  
School of Informatics  
University of Edinburgh  
Edinburgh EH8 9AB, UK  
t.maxwell@ed.ac.uk

**Jon Oberlander**  
School of Informatics  
University of Edinburgh  
Edinburgh EH8 9AB, UK  
j.oberlander@ed.ac.uk

**W. Bruce Croft**  
Dept. of Computer Science  
University of Massachusetts  
Amherst, MA 01003, USA  
croft@cs.umass.edu

## Abstract

Techniques that compare short text segments using dependency paths (or simply, paths) appear in a wide range of automated language processing applications including question answering (QA). However, few models in ad hoc information retrieval (IR) use paths for document ranking due to the prohibitive cost of parsing a retrieval collection. In this paper, we introduce a flexible notion of paths that describe chains of words on a dependency path. These chains, or *catenae*, are readily applied in standard IR models. Informative *catenae* are selected using supervised machine learning with linguistically informed features and compared to both non-linguistic terms and *catenae* selected heuristically with filters derived from work on paths. Automatically selected *catenae* of 1-2 words deliver significant performance gains on three TREC collections.

## 1 Introduction

In the past decade, an increasing number of techniques have used complex and effective syntactic and semantic features to determine the similarity, entailment or alignment between short texts. These approaches are motivated by the idea that sentence meaning can be flexibly captured by the syntactic and semantic relations between words, and encoded in dependency parse tree fragments. Dependency paths (or simply, paths) are compared using techniques such as tree edit distance (Punyakonok et al., 2004; Heilman and Smith, 2010), relation probability (Gao et al., 2004) and parse tree alignment (Wang et al., 2007; Park et al., 2011).

Much work on sentence similarity using dependency paths focuses on question answering (QA) where textual inference requires attention to linguistic detail. Dependency-based techniques can also be highly effective for ad hoc information

retrieval (IR) (Park et al., 2011). However, few path-based methods have been explored for ad hoc IR, largely because parsing large document collections is computationally prohibitive.

In this paper, we explore a flexible application of dependency paths that overcomes this difficulty. We reduce paths to chains of words called *catenae* (Osborne and Groß, 2012) that capture salient semantic content in an underspecified manner. *Catenae* can be used as lexical units in a reformulated query to explicitly indicate important word relationships while retaining efficient and flexible proximity matching. Crucially, this does not require parsing documents. Moreover, *catenae* are compatible with a variety of existing IR models.

We hypothesize that *catenae* identify most units of salient knowledge in text. This is because they are a condition for ellipsis, in which salient knowledge can be successfully omitted from text (Osborne and Groß, 2012). To our knowledge, this paper is the first time that *catenae* are proposed as a means for term selection in IR, and where ellipsis is considered as a means for identification of semantic units.

We also extend previous work with development of a linguistically informed, supervised machine learning technique for selection of informative *catenae*. Previous heuristic filters for dependency paths (Lin and Pantel, 2001; Shen et al., 2005; Cui et al., 2005) can exclude informative relations. Alternatively, treating all paths as equally informative (Punyakonok et al., 2004; Park et al., 2011; Moschitti, 2008) can generate noisy word relations and is computationally intensive.

The challenge of path selection is that no explicit information in text indicates which paths are relevant. Consider the *catenae* captured by heuristic filters for the TREC<sup>1</sup> query, ‘*What role does blood-alcohol level play in automobile accident fatalities*’ (#358, Table 1). It may appear obvious that the component words of ‘*role play*’

<sup>1</sup>Text REtrieval Conference, see <http://trec.nist.gov/>

Query: What role does blood-alcohol level play in automobile* accident fatalities*? (*abbreviated to `auto`, `fatal`)				
Catenaes	Governor-dependent	Predicate-argument	Nominal end slots	Sequential dependence
blood alcohol level play auto accident accident fatal role play play fatal blood alcohol play play accident fatal auto accident fatal level play fatal role play fatal role level play	blood alcohol level play auto accident accident fatal role play play fatal	auto accident accident fatal  play fatal  play accident fatal auto accident fatal	auto accident accident fatal   auto accident fatal level play fatal role play fatal	blood alcohol level play auto accident accident fatal       role blood alcohol level play auto

Table 1: Catenaes derived from dependency paths, as selected by heuristic methods. Selections are compared to sequential bigrams that use no linguistic knowledge.

and ‘*level play*’ do not have an important semantic relationship relative to the query, yet these catenae are described by parent-child relations that are commonly used to filter paths in text processing applications. Alternative filters that avoid such trivial word combinations also omit descriptions of key entities such as ‘*blood alcohol*’, and identify longer catenae that may be overly restrictive. These shortcomings suggest that an optimized selection process may improve performance of techniques that use dependency paths in ad hoc IR.

We identify three previously proposed selection methods, and compare them on the task of catenae selection for ad hoc IR. Selections are tested using three TREC collections: Robust04, WT10G, and GOV2. This provides a diverse platform for experiments. We also develop a linguistically informed machine learning technique for catenae selection that captures both key aspects of heuristic filters, and novel characteristics of catenae and paths. The basic idea is that selection, or weighting, of catenae can be improved by features that are *specific to paths*, rather than *generic for all terms*.

Results show that our selection method is more effective in identifying key catenae compared to previously proposed filters. Integration of the identified catenae in queries also improves IR effectiveness compared to a highly effective baseline that uses sequential bigrams with no linguistic knowledge. This model represents the obvious alternative to catenae for term selection in IR.

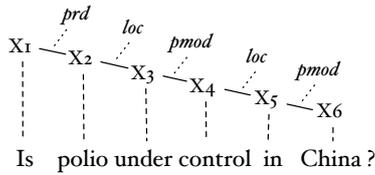
The rest of this paper is organised as follows. §2 reviews related work, §3 describes catenae and their linguistic motivation and §4 describes our selection method. §5 evaluates classification experiments using the supervised filter. §6 presents the results of experiments in ad hoc IR. Finally, §7 concludes the paper.

## 2 Related work

Techniques that compare short text segments using dependency paths are applied to a wide range of automated language processing tasks, including paraphrasing, summarization, entailment detection, QA, machine translation and the evaluation of word, phrase and sentence similarity. A generic approach uses a matching function to compare a dependency path between any two stemmed terms  $x$  and  $y$  in a sentence  $A$  with any dependency path between  $x$  and  $y$  in sentence  $B$ . The match score for  $A$  and  $B$  is computed over all dependency paths in  $A$ .

In QA this approach improves question representation, answer selection and answer ranking compared to methods that use bag-of-words and ngram features (Surdeanu et al., 2011). For example, Lin and Pantel (2001) present a method to derive paraphrasing rules for QA using analysis of paths that connect two nouns; Echihabi and Marcu (2003) align all paths in questions with trees for heuristically pruned answers; Cui et al. (2005) score answers using a variation of the IBM translation model 1; Wang et al. (2007) use quasi-synchronous translation to map all parent-child paths in a question to any path in an answer; and Moschitti (2008) explores syntactic and semantic kernels for QA classification.

In ad hoc IR, most models of term dependence use word co-occurrence and proximity (Song and Croft, 1999; Metzler and Croft, 2005; Srikanth and Srihari, 2002; van Rijsbergen, 1993). Syntactic language models for IR are a significant departure from this trend (Gao et al., 2004; Lee et al., 2006; Cai et al., 2007; Maisonnasse et al., 2007) that use dependency paths to address long-distance dependencies and normalize spurious differences in surface text. Paths are constrained in both



Catena (stoplisted)	Dependency paths
polio	<i>loc</i> <i>pmod</i>
polio control	polio → under → control
control	<i>loc</i> <i>pmod</i>
control China	control → in → China
China	<i>loc</i> <i>pmod</i> <i>loc</i> <i>pmod</i>
polio control China	polio → under → control → in → China

Figure 1: Catenae are an economical and intuitive representation of dependency paths.

queries and documents to parent-child relations. In contrast, (Park et al., 2011) present a quasi-synchronous translation model for IR that does not limit paths. This is based on the observation that semantically related words have a variety of direct and indirect relations. All of these models require parsing of an entire document collection.

Techniques using dependency paths in both QA and ad hoc IR show promising results, but there is no clear understanding of which path constraints result in the greatest IR effectiveness. We directly compare selections of catenae as a simplified representation of paths.

In addition, a vast number of methods have been presented for term weighting and selection in ad hoc IR. Our supervised selection extends the successful method presented by Bendersky and Croft (2008) for selection and weighting of query noun phrases (NPs). It also extends work for determining the variability of governor-dependent pairs (Song et al., 2008). In contrast to this work, we apply linguistic features that are specific to catenae and dependency paths, and select among units containing more than two content-bearing words.

### 3 Catenae as semantic units

Catenae (Latin for ‘chain’, singular *catena*) are dependency-based syntactic units. This section outlines their unique semantic properties.

A catena is defined on a dependency graph that has lexical nodes (or words) linked by binary asymmetrical relations called dependencies. Dependencies hold between a *governor* and a *dependent* and may be syntactic or semantic in nature (Nivre, 2005). A dependency graph is usually acyclic such that each node has only one governor, and one root node of the tree does not depend on any other node.

A catena is a word, or sequence of words that are continuous with respect to a walk on a dependency

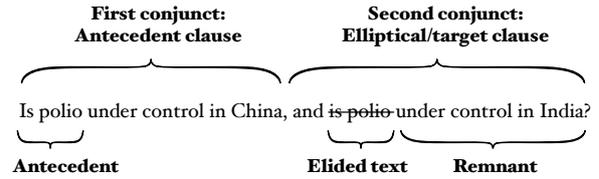


Figure 2: Ellipsis in a coordinated construct.

graph. For example, Fig. 1 shows a dependency parse that generates 21 catenae in total: (using  $i$  for  $X_i$ ) 1, 2, 3, 4, 5, 6, 12, 23, 34, 45, 56, 123, 234, 345, 456, 1234, 2345, 3456, 12345, 23456, 123456. We process catenae to remove stop words on the INQUERY stoplist (Allan et al., 2000) and lexical units containing 18 TREC description stop words such as ‘describe’. This results in a reduced set of catenae as shown in Fig. 1.

A dependency path is ordered and includes both word tokens and the relations between them. In contrast, a catena is a set of word types that may be ordered or partially ordered. A catena is an economical, intuitive lexical unit that corresponds to a dependency path and is argued to play an important role in syntax (Osborne et al., 2012).

In this paper, we explore catenae instead of paths for ad hoc IR due to their suitability for efficient IR models and flexible representation of language semantics. Specifically, we note that catenae identify words that can be omitted in elliptical constructions (Osborne et al., 2012). They thus represent *salient semantic information* in text. To clarify this insight, we briefly review catenae in ellipsis.

#### 3.1 Semantic units in ellipsis

Fig. 2 shows terminology for the phenomenon of ellipsis. The omitted words are called *elided* text, and words that could be omitted, but are not, we call *elliptical candidates*.

Ellipsis relies on the logical structure of a coordinated construction in which two or more elements, such as sentences, are joined by a conjunctive word or phrase such as ‘and’ or ‘more than’. A coordinated structure is required because the omitted words are ‘filled in’ by assuming a parallel relation  $p$  between the first and second conjunct. In ellipsis,  $p$  is omitted and its arguments are retained in text. In order for ellipsis to be successful and grammatically correct,  $p$  must be salient shared knowledge at the time of communication (Prince, 1986; Steedman, 1990). If  $p$  is salient then the omitted text can be inferred. If  $p$  is not salient then the omission of words merely results in ungrammatical, or incoherent, sentences.

This framework is practically illustrated in Fig.



Nugues, 2008) to generate the dependency parse. This enables us to explore semantic classification features and is highly accurate. However, any dependency parser may be applied instead. For comparison, catenae extracted from 500 queries using the Stanford dependency parser (de Marneffe et al., 2006) overlap with 77% of catenae extracted from the same queries using the applied parser.

#### 4.1 Feature Classes

Four feature classes are presented in Table 2:

**Ellipsis candidates:** The ellipsis hypothesis suggests that informative catenae are elliptical candidates. However, queries are not in the coordinated structures required for ellipsis. To enable extraction of characteristic features we (a) construct a *coordinated query* by adding the query to itself; and (b) elide catenae from the second conjunct. For example, for the query, *Is polio under control in China?* we have:

- (a) Is polio under control in China, and is polio under control in China?
- (b) Is polio under control in China, and is polio in China?

We refer to the words in (b) as the *query remainder* and use this to identify features detailed in Table 2.

**Dependency path features:** Part-of-speech tags and semantic roles have been used to filter dependency paths. We identify several features that use these characteristics from prior work (Table 2).

In addition, variability in the separation distance in documents observed for words that have governor-dependent relations in queries has been proposed for identification of promising paths (Song et al., 2008). We also observe that due to the phenomenon of rising, words that form catenae can be discontinuous in text, and the ability of catenae to match similar word combinations is limited by variability of how they appear in documents. Thus, we propose features for separation distance, but use efficient collection statistics rather than summing statistics for every document in a collection.

**Co-occurrence features:** A governor  $w_1$  tends to subcategorize for its dependents  $w_n$ . This means that  $w_1$  often determines the choice of  $w_n$ . We conclude that co-occurrence is an important feature of dependency relations (Mel'čuk, 2003). In addition, term frequencies and inverse document frequencies calculated using word co-occurrence measures are commonly used in IR. We use features previously proposed for filtering terms in IR (Bendersky and Croft, 2008) with two methods

to normalize co-occurrence counts for catenae of different lengths: a factor  $|c|^{-|c|}$ , where  $|c|$  is the number of words in catena  $c$  (Hagen et al., 2011), and the average score for a feature type over all pairwise word combinations in  $c$ .

**IR performance predictors:** Catenae take the same form as typical IR search terms. For this reason, we also use predictors of IR effectiveness previously applied to IR terms.

In general, path and co-occurrence features are similar to those applied by Surdeanu et al. (2011) but we do not parse documents. Path features are also similar to Song et al. (2008), but more efficient and suited to units of variable length. Ellipsis features have not been used before.

## 5 Experimental setup

### 5.1 Classification

Catenae selection is framed as a supervised classification problem trained on binary human judgments of *informativeness*: how well catenae represent a query and discriminate between relevant and non-relevant documents in a collection. Kappa for two annotators on catenae in 100 sample queries was 0.63, and test-retest reliability for individual judges was similar  $(0.62)^3$ . Although this is low, human annotations produced consistently better classification accuracy than other labelling methods explored.

We use the Weka (Hall et al., 2009) Adaboost.M1 meta-classifier (Freund and Schapire, 1996) with unpruned C4.5 decision trees as base learners to classify catenae as informative or not. Adaboost.M1 boosts decisions over  $T$  weak learners for  $T$  features using weighted majority voting. At each round, predictions of a new learner are focused on incorrectly classified examples from the previous round. Adaboost.M1 was selected in preference to other algorithms because it performed better in preliminary experiments, leverages many weak features to advantage, and usually does not overfit (Schapire et al., 1997).

Predictions are made using 10-fold cross-validation. There are roughly three times the number of uninformative catenae compared to informative catenae. In addition, the number of training examples is small (1295 to 5163 per collection). To improve classifier accuracy, the training data for each collection is supplemented and balanced by generating examples from queries for

<sup>3</sup>Catenae, judgments and annotation details available at [ciir.cs.umass.edu/~tmaxwell](http://ciir.cs.umass.edu/~tmaxwell)

Ellipsis candidate features (E)	
<i>R_ppl1</i>	Minimum perplexity of ngrams with length 2, 3, and 4 in a window of up to a 3 words around the site of catenae omission. This is the area where ungrammaticality may be introduced. For the remainder R='ABCDE&ABE' we compute ppl1 for {&ABE, &AB, ABE, &A, AB, BE}.
<i>R_strict</i>	Compliance with strict hand-coded rules for grammaticality of a remainder. Rules include unlikely orderings of punctuation and part-of-speech (POS) tags (e.g. ,, ), poor placement of determiners and punctuation, and orphaned words, such as adjectives without the nouns they modify.
<i>R_relax</i>	A relaxed version of hand-coded rules for <i>R_strict</i> . Some rules were observed to be overly aggressive in detection of ungrammatical remainders.
<i>NP_split</i>	Unsuccessful ellipsis often results if elided words only partly describe a base NP. Boolean feature for presence of a partial NP in the remainder. NPs (and PPs) are identified using the MontyLingua toolkit.
<i>PP_split</i>	As for <i>NP_split</i> , defined for prepositional phrases (PP).
<i>F_split</i>	As for <i>NP_split</i> , defined for finite clauses.
Dependency path features (D)	
<i>c_ppl1</i>	Dependency paths traverse nodes including stopwords and may be filtered based on POS tags. We use perplexity for the sequence of POS tags in catenae before removing stopwords. This is computed using a POS language model built on ukWaC parsed wikipedia data (Baroni et al., 2009).
<i>phClass</i>	Phrasal class for a catena, with options <i>NP</i> , <i>VP</i> and <i>Other</i> . A catena has a NP or VP class if it is, or is entirely contained by, an NP or VP (Song et al., 2008).
<i>semRole</i>	Boolean feature indicating whether a catena describes all, or part of, a predicate-argument structure (PAS). Previous work approximated PAS by using paths between head nouns and verbs, and all paths excluding those within base chunks.
<i>nomEnd</i>	Boolean indicating whether the words at each end of the catena are nouns (or the catena is a single noun).
<i>sepMode</i>	Most frequent separation distance of words in catena <i>c</i> in the retrieval collection, with possible values $S = \{1, 2, 3, long\}$ . 1 means that all words are adjacent, 2 means separation by 0-1 words, and <i>long</i> means containment in a window of size $4 *  c $ .
<i>H_c</i>	Entropy for separation distance <i>s</i> of words in catena <i>c</i> in the retrieval collection. $f_s$ is the frequency of <i>c</i> in window size <i>s</i> , and $f_S$ is the frequency of <i>c</i> in a window of size $4 *  c $ . All $f$ are normalized for catena length using $ c ^{ c }$ (Hagen et al., 2011). $H_c = \sum_{s \in S} \frac{f_s + 0.5}{f_S + 0.5} \log_2 \frac{f_s + 0.5}{f_S + 0.5}$

Dependency path features (D) (continued)	
<i>sepRatio</i>	Where $f_s$ and $f_S$ are defined as for <i>H_c</i> : $sepRatio_c = \frac{f_{s>2} + 0.5}{f_S + 0.5}$
<i>wRatio</i>	For words <i>w</i> in catena <i>c</i> ; $f_S$ is defined as for <i>H_c</i> . $wRatio_c = \frac{0.5 + \frac{1}{ c } \sum_{w \in c} f_w}{f_S + 0.5}$
Co-occurrence features (C)	
<i>isSeq</i>	Boolean indicating if catena words are sequential in stoplisted surface text.
<i>cf_ow</i>	Frequency of a catena in the retrieval collection, words appearing ordered in a window the length of the catena.
<i>cf_uw</i>	As for <i>cf_ow</i> , but words may appear unordered.
<i>cf_uw8</i>	As for <i>cf_uw</i> , but the window has a length of 8 words.
<i>idf_ow</i>	Inverse document frequency ( <i>idf</i> ) where document frequency ( <i>df</i> ) of a catena is calculated using <i>cf_ow</i> windows. Let <i>N</i> be the number of documents in the retrieval collection, then: $idf(C_i) = \log_2 \frac{N}{df(C_i)}$ and $idf(C_i) = N$ if $df(C_i) = 0$ .
<i>idf_uw</i>	As for <i>idf_ow</i> , but words may appear unordered.
<i>idf_uw8</i>	As for <i>idf_uw</i> , but the window has a length of 8 words.
<i>gf</i>	Google ngrams frequency (Brants and Franz, 2006) from a web crawl of approximately one trillion English word tokens. Counts from a large collection are expected to be more reliable than those from smaller test collections.
<i>qf_in</i>	Frequency of appearance in queries from the Live Search 2006 search query log (approximately 15 million queries). Query log frequencies are a measure of the likelihood that a catena will appear in any query.
<i>wf_in</i>	As for <i>qf_in</i> , but using frequency counts in Wikipedia titles instead of queries.
IR performance prediction features (I)	
<i>c_len</i>	Length of a stopped catenae. Longer terms tend to reduce IR recall.
<i>WIG</i>	Normalized Weighted Information Gain (WIG) is the change in information over top ranked documents between a random ranked list and an actual ranked list retrieved with a catena <i>c</i> (Zhou and Croft, 2007). $wig(c) = \frac{\frac{1}{k} \sum_{d \in D_k(c)} \log p(c d) - \log p(c C)}{-\log p(c C)}$ where $D_k$ are the top $k=50$ documents retrieved with catena <i>c</i> from collection <i>C</i> , and $p(c \cdot)$ are maximum likelihood estimates. A second feature uses the average WIG score for all pairwise word combinations in <i>c</i> .

Table 2: Classifier features.

	Feature Classes							
	E-D-CI		E-D		E-CI		D-CI	
	Pr	R	Pr	R	Pr	R	Pr	R
ROB04	86.2	72.8	83.5	67.5	86.2	71.7	86.2	72.0
WT10G	79.3	67.1	76.9	59.7	77.2	65.6	79.6	66.1
GOV2	77.0	68.0	70.9	61.8	72.8	63.9	75.5	67.2

Table 3: Average classifier precision (Pr) and recall (R) over 10 folds. Pr is % positive predictions that are correct. R is % positive labeled instances predicted as positive. A combination of all classes marginally performs best.

other collections used in this paper, plus TREC8-QA. For example, training data for Robust04 includes data from WT10G, GOV2 and TREC8-QA. Any examples that replicate catenae in the test collection are excluded. For Robust04, WT10G and GOV2 respectively, 30%, 82% and 69% of the training data is derived from other collections.

## 5.2 Classification results

Average classification precision and recall is shown in Table 3. Co-occurrence and IR effectiveness prediction features (CI) was the most influential class, and accounted for 70% of all features in the model. Performance is marginally better using all features (E-D-CI) with a moderate improvement over human agreement on the annotation task. The E-D-CI filter is used in subsequent experiments.

Catenae were predicted for all queries. Predictions were more accurate for Robust04 than the other two collections. One potential explanation is that Robust04 queries are longer on average (up to 32 content words per query, compared to up to 16 words) so they generate a more diverse set of catenae that are more easily distinguished with respect to informativeness. The proportion of training data specific to the retrieval collection may also be a factor. Longer queries produce a greater number of catenae, so less training data from other collections is required.

## 6 Evaluation framework

### 6.1 Baseline IR models

Baselines are a unigram query likelihood (QL) model (bag of words) and a highly effective sequential dependence (SD) variant of the Markov random field (MRF) model (Metzler and Croft, 2005). SD uses a linear combination of three cliques of terms, where each clique is prioritized by a weight  $\lambda_c$ . The first clique contains individual words (query likelihood  $QL$ ),  $\lambda_1 = 0.85$ . The second clique contains query bigrams that match

document bigrams in 2-word ordered windows ( $\#l^1$ ),  $\lambda_2 = 0.1$ . The third clique uses the same bigrams as clique 2 with an 8-word unordered window ( $\#uw8^8$ ),  $\lambda_3 = 0.05$ . For example, the query *new york city* in Indri<sup>4</sup> query language is:

```
#weight(
 $\lambda_1$  #combine(new york city)
 $\lambda_2$  #combine(#1(new york) #1(york city))
 $\lambda_3$  #combine(#uw8(new york) #uw8(york city)))
```

SD is a competitive baseline in IR (Bendersky and Croft, 2008; Park et al., 2011; Xue et al., 2010). Our reformulated model uses the same query format as SD, but the second and third cliques contain filtered catenae instead of query bigrams. In addition, because catenae may be multi-word units, we adjust the unordered window size to  $4 * |c|$ . So, if two catenae ‘*york*’ and ‘*new york city*’ are selected, the last clique has the form:

```
 $\lambda_3$  #combine( york #uw12(new york city))
```

This query representation enables word relations to be explicitly indicated while maintaining efficient and flexible matching of catenae in documents. Moreover, it does not use dependency relations between words during retrieval, so there is no need to parse a collection.

### 6.2 Baseline catenae selection

We explore four filters for catenae. Three are based on previous work and describe heuristic features of promising catenae. The fourth is our novel supervised classifier.

**NomEnd:** Catenae starting and ending with nouns, or containing only one word that is a noun. Paths between nouns are used by Lin and Pantel (2001).

**SemRol:** Catenae in which all component words are either predicates or argument heads. This is based on work that uses paths between head nouns and verbs (Shen et al., 2005), semantic roles (Moschitti, 2008), and all dependency paths except those that occur between words in the same base chunk (e.g. noun / verb phrase) (Cui et al., 2005).

**GovDep:** Catenae containing words with a governor-dependent relation. Many IR models use this form of path filtering e.g. (Gao et al., 2004; Wang et al., 2007). Relations are ‘collapsed’ by removing stopwords to reduce the distance between content nodes in a dependency graph.

<sup>4</sup><http://www.lemurproject.org/>

	ROBUST04		WT10G		GOV2	
	MAP	R-Pr	MAP	R-Pr	MAP	R-Pr
QL	25.25	28.69	19.55	22.77	25.77	31.26
SD	26.57†	30.02‡	20.63	24.31†	28.00†	33.30†
NomEnd	25.91†	29.35‡	20.81†	24.27†	27.41†	32.94†
GovDep	26.26†	29.63†	21.06	24.23†	27.87†	33.51†
SemRol	25.70†	29.06	19.78	22.93	26.76	32.49†
SFeat	<b>27.04†</b>	30.11†	20.84†	24.31†	28.43†	33.84†
SF-12	27.03†	<b>30.20†</b>	<b>21.62†</b>	<b>24.81†</b>	<b>28.57†</b>	<b>34.01†</b>

Table 4: IR results using filtered catenae consistently improve over non-linguistic methods. Significance( $p < .05$ ) shown compared to QL (†) and SD (‡).

	ROBUST04		WT10G		GOV2	
	MAP	R-Pr	MAP	R-Pr	MAP	R-Pr
SF-12	<b>27.03</b>	30.20	<b>21.62</b>	<b>24.81</b>	28.57	34.01
SF-123	26.83	<b>30.34</b>	21.34	24.64	<b>28.77</b>	<b>34.24</b>
SF-NE	26.51	29.86	21.42	24.55	27.96	33.26
SF-GD	26.22	29.48	20.33	23.72	28.30	33.83
Gold	27.92	31.15	22.56	25.69	29.65	35.08

Table 5: Results with supervised selection of catenae with specified length (SF-12, SF-123) are more effective than combinations of SFeat with heuristic NomEnd (SF-NE) or GovDep (SF-GD).

### 6.3 Experiments

Experiments compare queries reformulated using catenae selected by baseline filters and our supervised selection method (SFeat) to SD and a bag-of-words model (QL). We also compare IR effectiveness of all catenae filtered using SFeat with approaches that combine SFeat with baseline filters. All models are implemented using the Indri retrieval engine version 4.12.

### 6.4 Results

Results in Table 4 show significant improvement in mean average precision (MAP) of queries using catenae compared to QL. Consistent improvements over SD are also demonstrated for supervised selection applied to all catenae (SFeat) and catenae with only 1-2 words (SF-12) across all collections (Table 5). Overall, changes are small and fairly robust, with one half to two thirds of all queries showing less than 10% change in MAP.

Unlike sFeat, other filters tend to decrease performance compared to SD. Governor-dependent relations for WT10G are an exception and we speculate that this is due to a negative influence of 3-word catenae for this collection. Manual inspection suggests that WT10G queries are short and have relatively simple syntactic structure (e.g. few PP attachment ambiguities). This means that 3-word catenae (in all models except GovDep) tend to include uninformative words, such as ‘*reasons*’ in ‘*fasting religious reasons*’. In contrast, 3-word cate-

nae in other collections tend to identify query sub-concepts or phrases, such as ‘*science plants water*’.

Classification results for catenae separated by length, such that the classifier for catenae with a specific length are trained on examples of catenae with the same length, confirm this intuition. The rejection rate for 3-word catenae is twice as high for WT10G as for other collections. It is also more difficult to distinguish informative 3-word catenae compared to catenae with 1-2 words. To assess the impact of classification accuracy on IR effectiveness, Table 5 shows results with oracle knowledge of annotator judgments.

The SF-12 model combines catenae predicted for lengths 1 and 2. Its strong performance across all collections suggests that most of the benefit derived from catenae in IR is found in governor-dependent and single word units, where single words are important (GovDep uses only 2-word catenae). Another major observation (Table 5) is that mixing baseline heuristic filters with a supervised approach is not as successful as supervised selection alone. In particular, performance decreases for filtered governor-dependent pairs. This suggests that some important word relations in GovDep and NomEnd are captured by triangulation.

Finally, we review selected catenae for queries that perform significantly better or worse than SD ( $> 75\%$  change in MAP). The best IR effectiveness occurs when selected catenae clearly focus on the most important aspect of a query. Poor perfor-

mance is caused by a lack of focus in a catenae set, even though selected catenae are reasonable, or an emphasis on words that are not central to the query. The latter can occur when words that are not essential to query semantics appear in many catenae due to their position in the dependency graph.

## 7 Conclusion

We presented a flexible implementation of dependency paths for long queries in ad hoc IR that does not require dependency parsing a collection. Our supervised selection technique for catenae addresses the need to balance a representation of language expressiveness with effective, efficient statistical methods. This is a core challenge in computational linguistics.

It is not possible to directly compare performance of our approach with ad hoc techniques in IR that parse a retrieval collection. However, we note that a recent result using query translation based on dependency paths (Park et al., 2011) reports 14% improvement over query likelihood (QL). Our approach achieves 7% improvement over QL on the same collection. We conclude that catenae do not replace path-based techniques, but may offer some insight into their application, and have particular value when it is not practical to parse target documents to determine text similarity.

## Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

## References

- James Allan, Margaret E. Connell, W. Bruce Croft, Fang-Fang Feng, David Fisher, and Xiaoyan Li. 2000. INQUERY and TREC-9. In *Proceedings of TREC-9*, pages 551–562.
- Michael Bendersky and W. Bruce Croft. 2008. Discovering key concepts in verbose queries. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 491–498, New York, NY, USA. ACM.
- Keke Cai, Jiajun Bu, Chun Chen, and Guang Qiu. 2007. A novel dependency language model for information retrieval. *Journal of Zhejiang University SCIENCE A*, 8(6):871–882.
- Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 400–407, New York, NY, USA. ACM.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC-2006*.
- Abdessamad Echihabi and Daniel Marcu. 2003. A noisy-channel approach to question answering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 16–23, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Yoav Freund and Robert E. Schapire. 1996. Experiments with a new boosting algorithm. In *ICML'96*, pages 148–156.
- Jianfeng Gao, Jian-Yun Nie, Guangyuan Wu, and Guihong Cao. 2004. Dependence language model for information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 170–177, New York, NY, USA. ACM.
- Matthias Hagen, Martin Potthast, Benno Stein, and Christof Bräutigam. 2011. Query segmentation revisited. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 97–106, New York, NY, USA. ACM.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: an update. *SIGKDD Explorations Newsletter*, 11:10–18, November.
- Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1011–1019, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of CoNLL 2008*, pages 183–187.
- Changki Lee, Gary Geunbae Lee, and Myung-Gil Jang. 2006. Dependency structure language model for information retrieval. In *In ETRI journal*, volume 28, pages 337–346.
- Dekang Lin and Patrick Pantel. 2001. DIRT - discovery of inference rules from text. In *Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD-01)*, pages 323–328, San Francisco, CA.

- Loïc Maisonnasse, Eric Gaussier, and Jean-Pierre Chevallet. 2007. Revisiting the dependence language model for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 695–696, New York, NY, USA. ACM.
- Igor A. Mel'čuk. 2003. Levels of dependency in linguistic description: Concepts and problems. In V. Agel, L. Eichinger, H.-W. Eroms, P. Hellwig, H. J. Heringer, and H. Lobin, editors, *Dependency and Valency. An International Handbook of Contemporary Research*, volume 1, pages 188–229. Walter De Gruyter, Berlin–New York.
- Donald Metzler and W. Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 472–479, New York, NY, USA. ACM.
- Alessandro Moschitti. 2008. Kernel methods, syntax and semantics for relational text categorization. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 253–262, New York, NY, USA. ACM.
- Joakim Nivre. 2005. Dependency grammar and dependency parsing. Technical report, Växjö University: School of Mathematics and Systems Engineering.
- Timothy Osborne and Thomas Groß. 2012. Constructions are catenae: Construction grammar meets dependency grammar. *Cognitive Linguistics*, 23(1):165–216.
- Timothy Osborne, Michael Putnam, and Groß. 2012. Catenae: Introducing a novel unit of syntactic analysis. *Syntax*, 15(4):354–396, December.
- Jae Hyun Park, W. Bruce Croft, and David A. Smith. 2011. A quasi-synchronous dependence model for information retrieval. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 17–26, New York, NY, USA. ACM.
- Ellen F. Prince. 1986. On the syntactic marking of presupposed open propositions. In *Proceedings of the 22nd Annual Meeting of the Chicago Linguistic Society*, pages 208–222.
- V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *Proceedings of AI and MATH Symposium 2004 (Special session: Intelligent Text Processing)*.
- Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. 1997. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proceedings of ICML*, pages 322–330.
- Dan Shen, Geert-Jan M. Kruijff, and Dietrich Klakow. 2005. Exploring syntactic relation patterns for question answering. In *Proceedings of the Second international joint conference on Natural Language Processing*, IJCNLP'05, pages 507–518, Berlin, Heidelberg. Springer-Verlag.
- Fei Song and W. Bruce Croft. 1999. A general language model for information retrieval. In *Proceedings of the 8th ACM international conference on Information and knowledge management*, CIKM '99, pages 316–321, New York, NY, USA. ACM.
- Young-In Song, Kyoung-Soo Han, Sang-Bum Kim, So-Young Park, and Hae-Chang Rim. 2008. A novel retrieval approach reflecting variability of syntactic phrase representation. *Journal of Intelligent Information Systems*, 31(3):265–286, December.
- Munirathnam Srikanth and Rohini Srihari. 2002. Biterm language models for document retrieval. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, pages 425–426, New York, NY, USA. ACM.
- Mark J. Steedman. 1990. Gapping as Constituent Coordination. *Linguistics and Philosophy*, 13(2):207–263, April.
- Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2):351–383, June.
- C. J. van Rijsbergen. 1993. A theoretical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation*, 33(2):106–119.
- Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic, June. Association for Computational Linguistics.
- Xiaobing Xue, Samuel Huston, and W. Bruce Croft. 2010. Improving verbose queries using subset distribution. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1059–1068, New York, NY, USA. ACM.

# Coordination Structures in Dependency Treebanks

Martin Popel, David Mareček, Jan Štěpánek, Daniel Zeman, Zdeněk Žabokrtský

Charles University in Prague, Faculty of Mathematics and Physics

Institute of Formal and Applied Linguistics (ÚFAL)

Malostranské náměstí 25, CZ-11800 Praha, Czechia

{popel|marecek|stepanek|zeman|zabokrtsky}@ufal.mff.cuni.cz

## Abstract

Paratactic syntactic structures are notoriously difficult to represent in dependency formalisms. This has painful consequences such as high frequency of parsing errors related to coordination. In other words, coordination is a pending problem in dependency analysis of natural languages. This paper tries to shed some light on this area by bringing a systematizing view of various formal means developed for encoding coordination structures. We introduce a novel taxonomy of such approaches and apply it to treebanks across a typologically diverse range of 26 languages. In addition, empirical observations on convertibility between selected styles of representations are shown too.

## 1 Introduction

In the last decade, dependency parsing has gradually been receiving visible attention. One of the reasons is the increased availability of dependency treebanks, be they results of genuine dependency annotation projects or converted automatically from previously existing phrase-structure treebanks.

In both cases, a number of decisions have to be made during the construction or conversion of a dependency treebank. The traditional notion of dependency does not always provide unambiguous solutions, e.g. when it comes to attaching functional words. Worse, dependency representation is at a loss when it comes to representing paratactic linguistic phenomena such as coordination, whose nature is symmetric (two or more conjuncts play the same role), as opposed to the head-modifier asymmetry of dependencies.<sup>1</sup>

<sup>1</sup>We use the term *modifier* (or *child*) for all types of dependent nodes including *arguments*.

The dominating solution in treebank design is to introduce artificial rules for the encoding of coordination structures within dependency trees using the same means that express dependencies, i.e., by using edges and by labeling of nodes or edges. Obviously, any tree-shaped representation of a coordination structure (CS) must be perceived only as a “shortcut” since relations present in coordination structures form an undirected cycle, as illustrated already by Tesnière (1959). For example, if a noun is modified by two coordinated adjectives, there is a (symmetric) coordination relation between the two conjuncts and two (asymmetric) dependency relations between the conjuncts and the noun.

However, as there is no obvious linguistic intuition telling us which tree-shaped CS encoding is better and since the degree of freedom has several dimensions, one can find a number of distinct conventions introduced in particular dependency treebanks. Variations exist both in topology (tree shape) and labeling. The main goal of this paper is to give a systematic survey of the solutions adopted in these treebanks.

Naturally, the interplay of dependency and coordination links in a single tree leads to serious parsing issues.<sup>2</sup> The present study does not try to decide which coordination style is the best from the parsing point of view.<sup>3</sup> However, we believe that our survey will substantially facilitate experiments in this direction in the future, at least by exploring and describing the space of possible candidates.

<sup>2</sup>CSs have been reported to be one of the most frequent sources of parsing errors (Green and Žabokrtský, 2012; McDonald and Nivre, 2007; Kübler et al., 2009; Collins, 2003). Their impact on quality of dependency-based machine translation can also be substantial; as documented on an English-to-Czech dependency-based translation system (Popel and Žabokrtský, 2009), 39% of serious translation errors which are caused by wrong parsing have to do with coordination.

<sup>3</sup>There might be no such answer, as different CS conventions might serve best for different applications or for different parser architectures.

The rest of the paper is structured as follows. Section 2 describes some known problems related to CS. Section 3 shows possible “styles” for representing CS. Section 4 lists treebanks whose CS conventions we studied. Section 5 presents empirical observations on CS convertibility. Section 6 concludes the paper.

## 2 Related work

Let us first recall the basic well-known characteristics of CSs.

In the simplest case of a CS, a coordinating conjunction joins two (usually syntactically and semantically compatible) words or phrases called conjuncts. Even this simplest case is difficult to represent within a dependency tree because, in the words of Lombardo and Lesmo (1998): *Dependency paradigms exhibit obvious difficulties with coordination because, differently from most linguistic structures, it is not possible to characterize the coordination construct with a general schema involving a head and some modifiers of it.*

Proper formal representation of CSs is further complicated by the following facts:

- CSs with more than two conjuncts (multi-conjunct CSs) exist and are frequent.
- Besides “private” modifiers of individual conjuncts, there are modifiers shared by all conjuncts, such as in “*Mary came and cried*”. Shared modifiers may appear alongside with private modifiers of particular conjuncts.
- Shared modifiers can be coordinated, too: “*big and cheap apples and oranges*”.
- Nested (embedded) coordinations are possible: “*John and Mary or Sam and Lisa*”.
- Punctuation (commas, semicolons, three dots) is frequently used in CSs, mostly with multi-conjunct coordinations or juxtapositions which can be interpreted as CSs without conjunctions (e.g. “*Don’t worry, be happy!*”).
- In many languages, comma or other punctuation mark may play the role of the main coordinating conjunction.
- The coordinating conjunction may be a multiword expression (“*as well as*”).
- Deficient CSs with a single conjunct exist.
- Abbreviations like “*etc.*” comprise both the

conjunction and the last conjunct.

- Coordination may form very intricate structures when combined with ellipsis. For example, a conjunct can be elided while its arguments remain in the sentence, such as in the following traditional example: “*I gave the books to Mary and the records to Sue.*”
- The border between paratactic and hypotactic surface means of expressing coordination relations is fuzzy. Some languages can use enclitics instead of conjunctions/prepositions, e.g. Latin “*Senatus Populusque Romanus*”. Purely hypotactic surface means such as the preposition in “*John with Mary*” occur too.<sup>4</sup>
- Careful semantic analysis of CSs discloses additional complications: if a node is modified by a CS, it might happen that it is the node itself (and not its modifiers) what should be semantically considered as a conjunct. Note the difference between “*red and white wine*” (which is synonymous to “*red wine and white wine*”) and “*red and white flag of Poland*”. Similarly, “*five dogs and cats*” has a different meaning than “*five dogs and five cats*”.

Some of these issues were recognized already by Tesnière (1959). In his solution, conjuncts are connected by vertical edges directly to the head and by horizontal edges to the conjunction (which constitutes a cycle in every CS). Many different models have been proposed since, out of which the following are the most frequently used ones:

- MS = Mel’čuk style used in the Meaning-Text Theory (MTT): the first conjunct is the head of the CS, with the second conjunct attached as a dependent of the first one, third conjunct under the second one, etc. Coordinating conjunction is attached under the penultimate conjunct, and the last conjunct is attached under the conjunction (Mel’čuk, 1988),
- PS = Prague Dependency Treebank (PDT) style: all conjuncts are attached under the coordinating conjunction (along with shared modifiers, which are distinguished by a special attribute) (Hajič et al., 2006),

<sup>4</sup>As discussed by Stassen (2000), all languages seem to have some strategy for expressing coordination. Some of them lack the paratactic surface means (the so called WITH-languages), but the hypotactic surface means are present almost always.

- SS = Stanford parser style:<sup>5</sup> the first conjunct is the head and the remaining conjuncts (as well as conjunctions) are attached under it.

One can find various arguments supporting the particular choices. MTT possesses a complex set of linguistic criteria for identifying the governor of a relation (see Mazziotta (2011) for an overview), which lead to MS. MS is preferred in a rule-based dependency parsing system of Lombardo and Lesmo (1998). PS is advocated by Štěpánek (2006) who claims that it can represent shared modifiers using a single additional binary attribute, while MS would require a more complex co-indexing attribute. An argumentation of Tratz and Hovy (2011) follows a similar direction: *We would like to change our [MS] handling of coordinating conjunctions to treat the coordinating conjunction as the head [PS] because this has fewer ambiguities than [MS]. . .*

We conclude that the influence of the choice of coordination style is a well-known problem in dependency syntax. Nevertheless, published works usually focus only on a narrow ad-hoc selection of few coordination styles, without giving any systematic perspective.

Choosing a file format presents a different problem. Despite various efforts to standardize linguistic annotation,<sup>6</sup> no commonly accepted standard exists. The primitive format used for CoNLL shared tasks is widely used in dependency parsing, but its weaknesses have already been pointed out (cf. Straňák and Štěpánek (2010)). Moreover, particular treebanks vary in their contents even more than in their format, i.e. each treebank has its own way of representing prepositions or different granularity of syntactic labels.

### 3 Variations in representing coordination structures

Our analysis of variations in representing coordination structures is based on observations from a set of dependency treebanks for 26 languages.<sup>7</sup>

<sup>5</sup>We use the already established MS-PS-SS distinction to facilitate literature overview; as shown in Section 3, the space of possible coordination styles is much richer.

<sup>6</sup>For example, TEI (TEI Consortium, 2013), PML (Hana and Štěpánek, 2012), SynAF (ISO 24615, 2010).

<sup>7</sup>The primary data sources are the following: *Ancient Greek*: Ancient Greek Dependency Treebank (Bamman and Crane, 2011), *Arabic*: Prague Arabic Dependency Treebank 1.0 (Smrž et al., 2008), *Basque*: Basque Dependency Treebank (larger version than CoNLL 2007 generously pro-

In accordance with the usual conventions, we assume that each sentence is represented by one dependency tree, in which each node corresponds to one token (word or punctuation mark). Apart from that, we deliberately limit ourselves to CS representations that have shapes of connected subgraphs of dependency trees.

We limit our inventory of means of expressing CSs within dependency trees to (i) tree topology (presence or absence of a directed edge between two nodes, Section 3.1), and (ii) node labeling (additional attributes stored inside nodes, Section 3.2).<sup>8</sup> Further, we expect that the set of possible variations can be structured along several dimensions, each of which corresponds to a certain simple characteristic (such as choosing the leftmost conjunct as the CS head, or attaching shared modifiers below the nearest conjunct). Even if it does not make sense to create the full Cartesian product of all dimensions because some values cannot be combined, it allows to explore the space of possible CS styles systematically.<sup>9</sup>

#### 3.1 Topological variations

We distinguish the following dimensions of topological variations of CS styles (see Figure 1):

**Family – configuration of conjuncts.** We divide the topological variations into three main groups, labeled as Prague (fP), Moscow (fM), and

vided by IXA Group) (Aduriz and others, 2003), *Bulgarian*: BulTreeBank (Simov and Osenova, 2005), *Czech*: Prague Dependency Treebank 2.0 (Hajič et al., 2006), *Danish*: Danish Dependency Treebank (Kromann et al., 2004), *Dutch*: Alpino Treebank (van der Beek and others, 2002), *English*: Penn TreeBank 3 (Marcus et al., 1993), *Finnish*: Turku Dependency Treebank (Haverinen et al., 2010), *German*: Tiger Treebank (Brants et al., 2002), *Greek (modern)*: Greek Dependency Treebank (Prokopidis et al., 2005), *Hindi, Bengali and Telugu*: Hyderabad Dependency Treebank (Husain et al., 2010), *Hungarian*: Szeged Treebank (Csendes et al., 2005), *Italian*: Italian Syntactic-Semantic Treebank (Montemagni and others, 2003), *Latin*: Latin Dependency Treebank (Bamman and Crane, 2011), *Persian*: Persian Dependency Treebank (Rasooli et al., 2011), *Portuguese*: Floresta sintá(c)tica (Afonso et al., 2002), *Romanian*: Romanian Dependency Treebank (Călăcean, 2008), *Russian*: Syntagrus (Boguslavsky et al., 2000), *Slovene*: Slovene Dependency Treebank (Džeroski et al., 2006), *Spanish*: AnCora (Taulé et al., 2008), *Swedish*: Talbanken05 (Nilsson et al., 2005), *Tamil*: TamilTB (Ramasamy and Žabokrtský, 2012), *Turkish*: METU-Sabancı Turkish Treebank (Atalay et al., 2003).

<sup>8</sup>Edge labeling can be trivially converted to node labeling in tree structures.

<sup>9</sup>The full Cartesian product of variants in Figure 1 would result in topological 216 variants, but only 126 are applicable (the inapplicable combinations are marked with “—” in Figure 1). Those 126 topological variants can be further combined with labeling variants defined in Section 3.2.

Main family	Prague family (code fP) [14 treebanks]	Moscow family (code fM) [5 treebanks]	Stanford family (code fS) [6 treebanks]
<b>Choice of head</b>			
Head on left (code hL) [10 treebanks]			
Head on right (code hR) [14 treebanks]			
Mixed head (code hM) [1 treebank]	A mixture of hL and hR		
<b>Attachment of shared modifiers</b>			
Shared modifier below the nearest conjunct (code sN) [15 treebanks]			
Shared modifier below head (code sH) [11 treebanks]			
<b>Attachment of coordinating conjunction</b>			
Coordinating conjunction below previous conjunct (code cP) [2 treebanks]	—		
Coordinating conjunction below following conjunct (code cF) [1 treebank]	—		
Coordinating conjunction between two conjuncts (code cB) [8 treebanks]	—		
Coordinating conjunction as the head (code cH) is the only applicable style for the Prague family [14 treebanks]	—	—	—
<b>Placement of punctuation</b>			
values pP [7 treebanks], pF [1 treebank] and pB [15 treebanks] are analogous to cP, cF and cB (but applicable also to the Prague family)			

Figure 1: Different coordination styles, variations in tree topology. Example phrase: “(lazy) dogs, cats and rats”. Style codes are described in Section 3.1.

Stanford (fS) families.<sup>10</sup> This first dimension distinguishes the configuration of conjuncts: in the Prague family, all the conjuncts are siblings governed by one of the conjunctions (or a punctuation fulfilling its role); in the Moscow family, the conjuncts form a chain where each node in the chain depends on the previous (or following) node; in the Stanford family, the conjuncts are siblings except for the first (or last) conjunct, which is the

<sup>10</sup>Names are chosen purely as a mnemonic device, so that Prague Dependency Treebank belongs to the Prague family, Mel’čuk style belongs to the Moscow family, and Stanford parser style belongs to the Stanford family.

head.<sup>11</sup>

**Choice of head – leftmost or rightmost.** In the Prague family, the head can be either the leftmost<sup>12</sup> (hL) or the rightmost (hR) conjunction or punctuation. Similarly, in the Moscow and Stanford families, the head can be either the leftmost (hL) or the rightmost (hR) conjunct. A third op-

<sup>11</sup>Note that for CSs with just two conjuncts, fM and fS may look exactly the same (depending on the attachment of conjunctions and punctuation as described below).

<sup>12</sup>For simplicity, we use the terms left and right even if their meaning is reversed for languages with right-to-left writing systems such as Arabic or Persian.

tion (hM) is to mix hL and hR based on some criterion, e.g. the Persian treebank uses hR for coordination of verbs and hL otherwise. For the experiments in Section 5, we choose the head which is closer to the parent of the whole CS, with the motivation to make the edge between CS head and its parent shorter, which may improve parser training.

**Attachment of shared modifiers.** Shared modifiers may appear before the first conjunct or after the last one. Therefore, it seems reasonable to attach shared modifiers either to the CS head (SH), or to the nearest (i.e. first or last) conjunct (SN).

**Attachment of coordinating conjunctions.** In the Moscow family, conjunctions may be either part of the chain of conjuncts (cB), or they may be put outside of the chain and attached to the previous (cP) or following (cF) conjunct. In the Stanford family, conjunctions may be either attached to the CS head (and therefore *between* conjuncts) (cB), or they may be attached to the previous (cP) or the following (cF) conjunct. The cB option in both Moscow and Stanford families, treats conjunctions in the same way as conjuncts (with respect to topology only). In the Prague family, there is just one option available (cH) – one of the conjunctions is the CS head while the others are attached to it.

**Attachment of punctuation.** Punctuation tokens separating conjuncts (commas, semicolons etc.) could be treated the same way as conjunctions. However, in most treebanks it is treated differently, so we consider it as well. The values pP, pF and pB are analogous to cP, cF and cB except that punctuation may be also attached to the conjunction in case of pP and pF (otherwise, a comma before the conjunction would be non-projectively attached to the member following the conjunction).

The three established styles mentioned in Section 2 can be defined in terms of the newly introduced abbreviations: PS = fPhRsHcHpB, MS = fMhLsNcBp?, and SS = fShLsNcBp?.<sup>13</sup>

### 3.2 Labeling variations

Most state-of-the-art dependency parsers can produce labeled edges. However, the parsers produce only one label per edge. To fully capture CSs, we need more than one label, because there are several aspects involved (see the initial assump-

<sup>13</sup>The question marks indicate that the original Mel'čuk and Stanford parser styles ignore punctuation.

tions in Section 3): We need to identify the coordinating conjunction (its POS tag might not be enough), conjuncts, shared modifiers, and punctuation that separates conjuncts. Besides that, there should be a label classifying the dependency relation between the CS and its parent.

Some of the information can be retrieved from the topology of the tree and the “main label” of each node, but not everything. The additional information can be attached to the main label, but such approach obscures the logical structure.

In the **Prague family**, there are two possible ways to label a conjunction and conjuncts:

Code dU (“dependency labeled at the upper level of the CS”). The dependency relation of the whole CS to its parent is represented by the label of the conjunction, while the conjuncts are marked with a special label for conjuncts (e.g. ccof in the Hyderabad Dependency Treebank).

Code dL (“lower level”). The CS is represented by a coordinating conjunction (or punctuation if there is no conjunction) with a special label (e.g. Coord in PDT). Subsequently, each conjunct has its own label that reflects the dependency relation towards the parent of the whole CS, therefore, conjuncts of the same CS can have different labels, e.g. “Who[*SUBJ*] and why[*ADV*] did it?”

Most Prague family treebanks use SH, i.e. shared modifiers are attached to the head (coordinating conjunction). Each child of the head has to belong to one of three sets: conjuncts, shared modifiers, and punctuation or additional conjunctions. In PDT, conjuncts, punctuation and additional conjunctions are recognized by specific labels. Any other children of the head are shared modifiers.

In the **Stanford and Moscow families**, one of the conjuncts is the head. In practice, it is never labeled as a conjunct explicitly, because the fact that it is a conjunct can be deduced from the presence of conjuncts among its children. Usually, the other conjuncts are labeled as conjuncts; conjunctions and punctuation also have a special label. This type of labeling corresponds to the dU type.

Alternatively (as found in the Turkish treebank, dL), all conjuncts in the Moscow chain have their own dependency labels and the fact that they are conjuncts follows from the COORDINATION labels of the conjunction and punctuation nodes between them.

To represent shared modifiers in the Stan-

ford and Moscow families, an additional label is needed again to distinguish between private and shared modifiers since they cannot be distinguished topologically. Moreover, if nested CSs are allowed, a binary label is not sufficient (i.e. “shared” versus “private”) because it also has to indicate which conjuncts the shared modifier belongs to.<sup>14</sup>

We use the following binary flag codes for capturing which CS participants are distinguished in the annotation: m01 = shared modifiers annotated; m10 = conjuncts annotated; m11 = both annotated; m00 = neither annotated.

#### 4 Coordination Structures in Treebanks

In this section, we identify the CS styles defined in the previous section as used in the primary treebank data sources; statistical observations (such as the amount of annotated shared modifiers) presented here, as well as experiments on CS-style convertibility presented in Section 5.2, are based on the normalized shapes of the treebanks as contained in the HamleDT 1.0 treebank collection (Zeman et al., 2012).<sup>15</sup>

Some of the treebanks were downloaded individually from the web, but most of them came from previously published collections for dependency parsing campaigns: six languages from CoNLL-2006 (Buchholz and Marsi, 2006), seven languages from CoNLL-2007 (Nivre et al., 2007), two languages from CoNLL-2009 (Hajič and others, 2009), three languages from ICON-2010 (Husain et al., 2010). Obviously, there is a certain risk that the CS-related information contained in the source treebanks was slightly biased by the properties of the CoNLL format upon conversion. In addition, many of the treebanks were natively dependency-based (cf. the 2nd column of Table 1), but some were originally based on constituents and thus specific converters to the CoNLL format had to be created (for instance, the Spanish phrase-structure trees were converted to dependencies using a procedure described by Civit et al. (2006); similarly, treebank-specific converters have been used for other languages). Again,

<sup>14</sup>This is not needed in Prague family where shared modifiers are attached to the conjunction provided that each shared modifier is shared by conjuncts that form a full subtree together with their coordinating conjunctions; no exceptions were found during the annotation process of the PDT.

<sup>15</sup>A subset of the treebanks whose license terms permit redistribution is available directly at <http://ufal.mff.cuni.cz/hamledt/>.

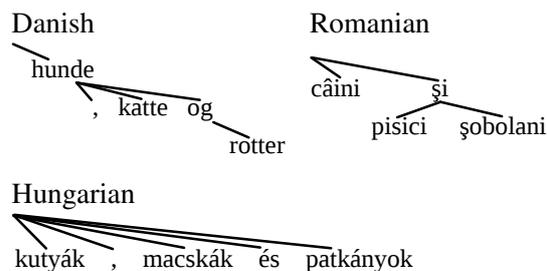


Figure 2: Annotation styles of a few treebanks do not fit well into the multidimensional space defined in Section 3.1.

there is some risk that the CS-related information contained in treebanks resulting from such conversions is slightly different from what was intended in the very primary annotation.

There are several other languages (e.g. Estonian or Chinese) which are not included in our study, despite of the fact that constituency treebanks do exist for them. The reason is that the choice of their CS style would be biased, because no independent converters exist – we would have to convert them to dependencies ourselves. We also know about several more dependency treebanks that we have not processed yet.

Table 1 shows 26 languages whose treebanks we have studied from the viewpoint of their CS styles. It gives the basic quantitative properties of the treebanks, their CS style in terms of the taxonomy introduced in Section 3, as well as statistics related to CSs: the average number of CSs per 100 tokens, the average number of conjuncts per one CS, the average number of shared modifiers per one CS,<sup>16</sup> and the percentage of nested CSs among all CSs. The reader can return to Figure 1 to see the basic statistics on the “popularity” of individual design decisions among the developers of dependency treebanks or constituency treebank converters.

CS styles of most treebanks are easily classifiable using the codes introduced in Section 3, plus a few additional codes:

- p0 = punctuation was removed from the treebank.

<sup>16</sup>All non-Prague family treebanks are marked sN and m00 or m10, (i.e. shared modifiers not marked in the original annotation, but attached to the head conjunct) because we found no counterexamples (modifiers attached to a conjunct, but not the nearest one). The HamleDT normalization procedure contains a few heuristics to detect shared modifiers, but it cannot recover the missing distinction reliably, so the numbers in the “SMs/CJ” column are mostly underestimated.

Language	Orig. type	Data set	Sents.	Tokens	Original CS style code	CSs / 100 tok.	CJs / CS	SMs / CS	Nested CS[%]	RT UAS
Ancient Greek	dep	prim.	31 316	461 782	fP hR sH cH pB dL m11	6.54	2.17	0.16	10.3	97.86
Arabic	dep	C07	3 043	116 793	fP hL sH cH pB dL m00	3.76	2.42	0.13	10.6	96.69
Basque	dep	prim.	11 225	151 593	fP hR sN cH pP dU m00	3.37	2.09	0.03	5.1	99.32
Bengali	dep	I10	1 129	7 252	fP hR sH cH pP dU m11	4.87	1.71	0.05	24.1	99.97
Bulgarian	phr	C06	13 221	196 151	fS hL sN cB pB dU m10	2.99	2.19	0.00	0.0	99.74
Czech	dep	C07	25 650	437 020	fP hR sH cH pB dL m11	4.09	2.16	0.20	14.6	99.42
Danish	dep	C06	5 512	100 238	fS* hL sN cP pB dU m10	3.68	1.93	0.13	7.5	99.76
Dutch	phr	C06	13 735	200 654	fP hR sN cH pP dU m10	2.06	2.17	0.05	3.3	99.47
English	phr	C07	40 613	991 535	fP hR sH cH pB dU m10	2.07	2.33	0.05	6.3	99.84
Finnish	dep	prim.	4 307	58 576	fS hL sN cB pB dU m10	4.06	2.41	0.00	6.4	99.70
German	phr	C09	38 020	680 710	fM hR sN cH pP dU m10	2.79	2.09	0.01	0.0	99.73
Greek	dep	C07	2 902	70 223	fP hR sH cH pB dL m11	3.25	2.48	0.18	7.2	99.43
Hindi	dep	I10	3 515	77 068	fP hR sH cH pP dU m11	2.45	1.97	0.04	10.3	98.35
Hungarian	phr	C07	6 424	139 143	fT hX sN cX pX dL m00	2.37	1.90	0.01	2.2	99.84
Italian	dep	C07	3 359	76 295	fS hL sN cB pB dU m10	3.32	2.02	0.03	3.8	99.51
Latin	dep	prim.	3 473	53 143	fP hR sH cH pB dL m11	6.74	2.24	0.41	12.3	97.45
Persian	dep	prim.	12 455	189 572	fM*hM sN cB pP dU m00	4.18	2.10	0.18	3.7	99.82
Portuguese	phr	C06	9 359	212 545	fS hL sN cH pB dU m10	2.51	1.95	0.26	11.1	99.16
Romanian	dep	prim.	4 042	36 150	fP* hR sN cH p0 dU m10	1.80	2.00	0.00	0.0	100.00
Russian	dep	prim.	34 895	497 465	fM hL sN cB p0 dU m10	4.02	2.02	0.07	3.9	99.86
Slovene	dep	C06	1 936	35 140	fP hR sH cH pB dL m00	4.31	2.49	0.00	10.8	98.87
Spanish	phr	C09	15 984	477 810	fS hL sN cB pB dU m10	2.79	1.98	0.14	12.7	99.24
Swedish	phr	C06	11 431	197 123	fM hL sN cF pF dU m10	3.94	2.19	0.13	0.7	99.66
Tamil	dep	prim.	600	9 581	fP hR sH cH pB dL m11	1.66	2.46	0.22	3.8	99.67
Telugu	dep	I10	1 450	5 722	fP hR sH cH pP dU m11	3.48	1.59	0.06	5.0	100.00
Turkish	dep	C07	5 935	69 695	fM hR sN cB pB dL m10	3.81	2.04	0.00	34.3	99.23

Table 1: Overview of analyzed treebanks. prim. = primary source; C06–C09 = CoNLL 2006–2009; I10 = ICON 2010; SM = shared modifier; CJ = conjunct; Nested CS = portion of CSs participating in nested CSs (both as the inner and outer CS); RT UAS = unlabeled attachment score of the roundtrip experiment described in Section 5. Style codes are defined in Sections 3 and 4.

- fM\* = Persian treebank uses a mix of fM and fS: fS for coordination of verbs and fM otherwise.

Figure 2 shows three other anomalies:

- fS\* = Danish treebank employs a mixture of fS and fM, where the last conjunct is attached indirectly via the conjunction.
- fP\* = Romanian treebank omits punctuation tokens and multi-conjunct coordinations get split.
- fT = Hungarian Szeged treebank uses “Tesnière family” – disconnected graphs for CSs where conjuncts (and conjunction and punctuation) are attached directly to the parent of CS, and so the other style dimensions are not applicable (hX, cX, pX).

## 5 Empirical Observations on Convertibility of Coordination Styles

The various styles cannot represent the CS-related information to the same extent. For example,

it is not possible to represent nested CSs in the Moscow and Stanford families without significantly changing the number of possible labels.<sup>17</sup> The dL style (which is most easily applicable to the Prague family) can represent coordination of different dependency relations. This is again not possible in the other styles without adding e.g. a special “prefix” denoting the relations.

We can see that the Prague family has a greater expressive power than the other two families: it can represent complex CSs using just one additional binary label, distinguishing between shared modifiers and conjuncts. A similar additional label is needed in the other styles to distinguish between shared and private modifiers.

Because of the different expressive power, converting a CS from one style to another may lead to a loss of information. For example, as

<sup>17</sup>Mel’čuk uses “grouping” to nest CSs – cf. related solutions involving coindexing or bubble trees (Kahane, 1997). However, these approaches were not used in any of the researched treebanks. To combine grouping with shared modifiers, each group in a tree should have a different identifier.

there is no way of representing shared modifiers in the Moscow family without an additional attribute, converting a CS with shared modifiers from Prague to Moscow family makes the modifiers private. When converting back, one can use certain heuristics to handle the most obvious cases, but sometimes the modifiers will stay private (very often, the nature of a modifier depends on context or is debatable even for humans, e.g. “*Young boys and girls*”).

### 5.1 Transformation algorithm

We developed an algorithm to transform one CS style to another. Two subtasks must be solved by the algorithm: identification of individual CSs and their participants, and transforming of the individual CSs.

Obviously, the individual CSs cannot be transformed independently because of coordination nesting. For instance, when transforming a nested coordination from the Prague style to the Moscow style (e.g. to fMhL), the leftmost conjunct in the inner (lower) coordination must climb up to become the head of the inner CS, but then it must climb up once again to become the head of the outer (upper) CS too. This shows that inner CSs must be transformed first.

We tackle this problem by a depth-first recursion. When going down the tree, we only recognize all the participants of the CSs, classify them and gather them in a separate data structure (one for each visited CS). The following four types of CS participants are distinguished: coordinating conjunctions, conjuncts, shared modifiers, and punctuations that separate conjuncts.<sup>18</sup> No change of the tree is performed during these descent steps.

When returning back from the recursion (i.e., when climbing from a node back up to its parent), we test whether the abandoned node is the topmost node of some CS. If so, then this CS is transformed, which means that its participants are rehanged and relabelled according to the target CS style.

This procedure naturally guarantees that the in-

<sup>18</sup>Conjuncts are explicitly marked in most styles. Coordinating conjunctions can be usually identified with the help of dependency labels and POS tags. Punctuation separating conjuncts can be detected with high accuracy using simple rules. If shared modifiers are not annotated (code m00 or m10), one can imagine rule-based heuristics or special classifiers trained to distinguish shared modifiers. For the experiments in this section, we use the HamleDT gold annotation attribute `is_shared_modifier`.

ner CSs are transformed first and that all CSs are transformed when the recursions returns to the root.

### 5.2 Roundtrip experiment

The number of possible conversion directions obviously grows quadratically with the number of styles. So far, we limited ourselves only to conversions from/to the style of the HamleDT treebank collection, which contains all the treebanks under our study already converted into a common scheme. The common scheme is based on the conventions of PDT, whose CS style is fPhRsHcHpB.<sup>19</sup>

We selected nine styles (3 families times 3 head choices) and transformed all the HamleDT scheme treebanks to these nine styles and back, which we call a *roundtrip*. Resulting averaged unlabeled attachment scores (UAS, evaluated against the HamleDT scheme) in the last column of Table 1 indicate that the percentage of transformation errors (i.e. tokens attached to a different parent after the roundtrip) is lower than 1% for 20 out of the 26 languages.<sup>20</sup> A manual inspection revealed two main error sources. First, as noted above, the Stanford and Moscow families have lower expressive power than the Prague family, so naturally, the inverse transformation was ambiguous and the transformation heuristics were not capable of identifying the correct variant every time. Second, we also encountered inconsistencies in the original treebanks (which we were not trying to fix in HamleDT for now).

## 6 Conclusions and Future Work

We described a (theoretically very large) space of possible representations of CSs within the dependency framework. We pointed out a range of details that make CSs a really complex phenomenon; anyone dealing with CSs in treebanking should take these observations into account.

We proposed a taxonomy of those approaches

<sup>19</sup>As documented in Zeman et al. (2012), the normalization procedures used in HamleDT embrace many other phenomena as well (not only those related to coordination), and involve both structural transformation and dependency relation labeling.

<sup>20</sup>Table 1 shows that Latin and Ancient Greek treebanks have on average more than 6 CSs per 100 tokens, more than 2 conjuncts per CS, and Latin has also the highest number of shared modifiers per CS. Therefore the percentage of nodes affected by the roundtrip is the highest for these languages and the lower roundtrip UAS is not surprising.

that have been argued for in literature or employed in real treebanks.

We studied 26 existing treebanks of different languages. For each value of each dimension in Figure 1, we found at least one treebank where the value is used; even so, several treebanks take their own unique path that cannot be clearly classified under the taxonomy (the taxonomy could indeed be extended, for the price of being less clearly arranged).

We discussed the convertibility between the various styles and implemented a universal tool that transforms between any two styles of the taxonomy. The tool achieves a roundtrip accuracy close to 100%. This is important because it opens the door to easily switching coordination styles for parsing experiments, phrase-to-dependency conversion etc.

While the focus of this paper is to explore and describe the expressive power of various annotation styles, we did not address the learnability of the styles by parsers. That will be a complementary point of view, and thus a natural direction of future work for us.

## Acknowledgments

We thank the providers of the primary data resources. The work on this project was supported by the Czech Science Foundation grants no. P406/11/1499 and P406/2010/0875, and by research resources of the Charles University in Prague (PRVOUK). This work has been using language resources developed and/or stored and/or distributed by the LINDAT-Clarin project of the Ministry of Education of the Czech Republic (project LM2010013). Further, we would like to thank Jan Hajič, Ondřej Dušek and four anonymous reviewers for many useful comments on the manuscript of this paper.

## References

Itzair Aduriz et al. 2003. Construction of a Basque dependency treebank. In *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories*.

Susana Afonso, Eckhard Bick, Renato Haber, and Diana Santos. 2002. “Floresta sintá(c)tica”: a treebank for Portuguese. In *LREC*, pages 1968–1703.

Nart B. Atalay, Kemal Oflazer, and Bilge Say. 2003. The annotation process in the Turkish treebank. In *Proceedings of the 4th Intern. Workshop on Linguistically Interpreted Corpora (LINC)*.

David Bamman and Gregory Crane. 2011. The Ancient Greek and Latin dependency treebanks. In *Language Technology for Cultural Heritage, Theory and Applications of Natural Language Processing*, pages 79–98. Springer Berlin Heidelberg.

Igor Boguslavsky, Svetlana Grigorieva, Nikolai Grigoriev, Leonid Kreidlin, and Nadezhda Frid. 2000. Dependency treebank for Russian: Concept, tools, types of information. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 987–991. Association for Computational Linguistics Morristown, NJ, USA.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

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

Montserrat Civit, Maria Antònia Martí, and Núria Bufí. 2006. Cat3LB and Cast3LB: From constituents to dependencies. In *FinTAL*, volume 4139 of *Lecture Notes in Computer Science*, pages 141–152. Springer.

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

Dóra Csendes, János Csirik, Tibor Gyimóthy, and András Kocsor. 2005. The Szeged treebank. In *TSD*, volume 3658 of *Lecture Notes in Computer Science*, pages 123–131. Springer.

Mihaela Călăcean. 2008. Data-driven dependency parsing for Romanian. Master’s thesis, Uppsala University, August.

Sašo Džeroski, Tomaž Erjavec, Nina Ledinek, Petr Pajas, Zdeněk Žabokrtský, and Andreja Žele. 2006. Towards a Slovene dependency treebank. In *LREC 2006*, pages 1388–1391, Genova, Italy. European Language Resources Association (ELRA).

Nathan Green and Zdeněk Žabokrtský. 2012. Hybrid combination of constituency and dependency trees into an ensemble dependency parser. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 19–26, Avignon, France. Association for Computational Linguistics.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, Zdeněk Žabokrtský, and Magda Ševčíková-Razímová. 2006. Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

- Jan Hajič et al. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*, June 4-5, Boulder, Colorado, USA.
- Jirka Hana and Jan Štěpánek. 2012. Prague markup language framework. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 12–21, Stroudsburg, PA, USA. Association for Computational Linguistics, Association for Computational Linguistics.
- Katri Haverinen, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Filip Ginter, and Tapio Salakoski. 2010. Treebanking Finnish. In *Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories (TLT9)*, pages 79–90.
- Samar Husain, Prashanth Mannem, Bharat Ambati, and Phani Gadde. 2010. The ICON-2010 tools contest on Indian language dependency parsing. In *Proceedings of ICON-2010 Tools Contest on Indian Language Dependency Parsing*, Kharagpur, India.
- ISO 24615. 2010. Language resource management – Syntactic annotation framework (SynAF).
- Sylvain Kahane. 1997. Bubble trees and syntactic representations. In *Proceedings of the 5th Meeting of the Mathematics of the Language, DFKI, Saarbrücken*.
- Matthias T. Kromann, Line Mikkelsen, and Stine Kern Lynge. 2004. Danish dependency treebank.
- Sandra Kübler, Erhard Hinrichs, Wolfgang Maier, and Eva Klett. 2009. Parsing coordinations. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 406–414, Athens, Greece, March. Association for Computational Linguistics.
- Vincenzo Lombardo and Leonardo Lesmo. 1998. Unit coordination and gapping in dependency theory. In *Processing of Dependency-Based Grammars; proceedings of the workshop. COLING-ACL*, Montreal.
- Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330.
- Nicolar Mazziotta. 2011. Coordination of verbal dependents in Old French: Coordination as a specified juxtaposition or apposition. In *Proceedings of International Conference on Dependency Linguistics (DepLing 2011)*.
- Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131.
- Igor A. Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press.
- Simonetta Montemagni et al. 2003. Building the Italian syntactic-semantic treebank. In *Building and using Parsed Corpora*, Language and Speech series, pages 189–210, Dordrecht. Kluwer.
- Jens Nilsson, Johan Hall, and Joakim Nivre. 2005. MAMBA meets TIGER: Reconstructing a Swedish treebank from antiquity. In *Proceedings of the NODALIDA Special Session on Treebanks*.
- Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL 2007 Shared Task. EMNLP-CoNLL*, June.
- Martin Popel and Zdeněk Žabokrtský. 2009. Improving English-Czech Tectogrammatical MT. *The Prague Bulletin of Mathematical Linguistics*, (92):1–20.
- Prokopis Prokopidis, Elina Desipri, Maria Koutsombogera, Harris Papageorgiou, and Stelios Piperidis. 2005. Theoretical and practical issues in the construction of a Greek dependency treebank. In *Proceedings of the 4th Workshop on Treebanks and Linguistic Theories (TLT)*, pages 149–160.
- Loganathan Ramasamy and Zdeněk Žabokrtský. 2012. Prague dependency style treebank for Tamil. In *Proceedings of LREC 2012*, pages 23–25, Istanbul, Turkey. European Language Resources Association.
- Mohammad Sadeh Rasooli, Amirsaeid Moloodi, Manouchehr Kouhestani, and Behrouz Minaei-Bidgoli. 2011. A syntactic valency lexicon for Persian verbs: The first steps towards Persian dependency treebank. In *5th Language & Technology Conference (LTC): Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 227–231, Poznań, Poland.
- Kiril Simov and Petya Osenova. 2005. Extending the annotation of BulTreeBank: Phase 2. In *The Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*, pages 173–184, Barcelona, December.
- Otakar Smrž, Viktor Bielický, Iveta Kouřilová, Jakub Kráčmar, Jan Hajič, and Petr Zemánek. 2008. Prague Arabic dependency treebank: A word on the million words. In *Proceedings of the Workshop on Arabic and Local Languages (LREC) 2008*, pages 16–23, Marrakech, Morocco. European Language Resources Association.
- Leon Stassen. 2000. And-languages and with-languages. *Linguistic Typology*, 4(1):1–54.
- Jan Štěpánek. 2006. *Capturing a Sentence Structure by a Dependency Relation in an Annotated Syntactical Corpus (Tools Guaranteeing Data Consistency)* (in Czech). Ph.D. thesis, Charles Univer-

sity in Prague, Faculty of Mathematics and Physics, Prague, Czech Republic.

Pavel Straňák and Jan Štěpánek. 2010. Representing layered and structured data in the CoNLL-ST format. In Alex Fang, Nancy Ide, and Jonathan Webster, editors, *Proceedings of the Second International Conference on Global Interoperability for Language Resources*, pages 143–152, Hong Kong, China. City University of Hong Kong, City University of Hong Kong.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel annotated corpora for Catalan and Spanish. In *LREC*. European Language Resources Association.

TEI Consortium. 2013. TEI P5: Guidelines for Electronic Text Encoding and Interchange.

Lucien Tesnière. 1959. *Éléments de syntaxe structurale*. Paris.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of EMNLP*, pages 1257–1268, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.

Leonor van der Beek et al. 2002. Chapter 5. The Alpino dependency treebank. In *Algorithms for Linguistic Processing NWO PIONIER Progress Report*, Groningen, The Netherlands.

Daniel Zeman, David Mareček, Martin Popel, Loganathan Ramasamy, Jan Štěpánek, Zdeněk Žabokrtský, and Jan Hajič. 2012. HamleDT: To parse or not to parse? In *Proceedings of LREC 2012*, pages 2735–2741, İstanbul, Turkey. European Language Resources Association.

# GlossBoot: Bootstrapping Multilingual Domain Glossaries from the Web

Flavio De Benedictis, Stefano Faralli and Roberto Navigli

Dipartimento di Informatica

Sapienza Università di Roma

flavio.debene@gmail.com, {faralli, navigli}@di.uniroma1.it

## Abstract

We present GlossBoot, an effective minimally-supervised approach to acquiring wide-coverage domain glossaries for many languages. For each language of interest, given a small number of hypernymy relation seeds concerning a target domain, we bootstrap a glossary from the Web for that domain by means of iteratively acquired term/gloss extraction patterns. Our experiments show high performance in the acquisition of domain terminologies and glossaries for three different languages.

## 1 Introduction

Much textual content, such as that available on the Web, contains a great deal of information focused on specific areas of knowledge. However, it is not infrequent that, when reading a domain-specific text, we humans do not know the meaning of one or more terms. To help the human understanding of specialized texts, repositories of textual definitions for technical terms, called glossaries, are compiled as reference resources within each domain of interest. Interestingly, electronic glossaries have been shown to be key resources not only for humans, but also in Natural Language Processing (NLP) tasks such as Question Answering (Cui et al., 2007), Word Sense Disambiguation (Duan and Yates, 2010; Faralli and Navigli, 2012) and ontology learning (Navigli et al., 2011; Velardi et al., 2013).

Today large numbers of glossaries are available on the Web. However most such glossaries are small-scale, being made up of just some hundreds of definitions. Consequently, individual glossaries typically provide a partial view of a given domain. Moreover, there is no easy way of retrieving the subset of Web glossaries which appertains to a domain of interest. Although online services such

as Google Define allow the user to retrieve definitions for an input term, such definitions are extracted from Web glossaries and put together for the given term regardless of their domain. As a result, gathering a large-scale, full-fledged domain glossary is not a speedy operation.

Collaborative efforts are currently producing large-scale encyclopedias, such as Wikipedia, which are proving very useful in NLP (Hovy et al., 2013). Interestingly, wikipedias also include manually compiled glossaries. However, such glossaries still suffer from the same above-mentioned problems, i.e., being incomplete or over-specific,<sup>1</sup> and hard to customize according to a user's needs.

To automatically obtain large domain glossaries, over recent years computational approaches have been developed which extract textual definitions from corpora (Navigli and Velardi, 2010; Reiplinger et al., 2012) or the Web (Fujii and Ishikawa, 2000). The former methods start from a given set of terms (possibly automatically extracted from a domain corpus) and then harvest textual definitions for these terms from the input corpus using a supervised system. Web-based methods, instead, extract text snippets from Web pages which match pre-defined lexical patterns, such as “X is a Y”, along the lines of Hearst (1992). These approaches typically perform with high precision and low recall, because they fall short of detecting the high variability of the syntactic structure of textual definitions. To address the low-recall issue, recurring cue terms occurring within dictionary and encyclopedic resources can be automatically extracted and incorporated into lexical patterns (Saggion, 2004). However, this approach is term-specific and does not scale to arbitrary terminologies and domains.

In this paper we propose GlossBoot, a novel approach which reduces human intervention to a bare minimum and exploits the Web to learn a

<sup>1</sup><http://en.wikipedia.org/wiki/Portal:Contents/Glossaries>

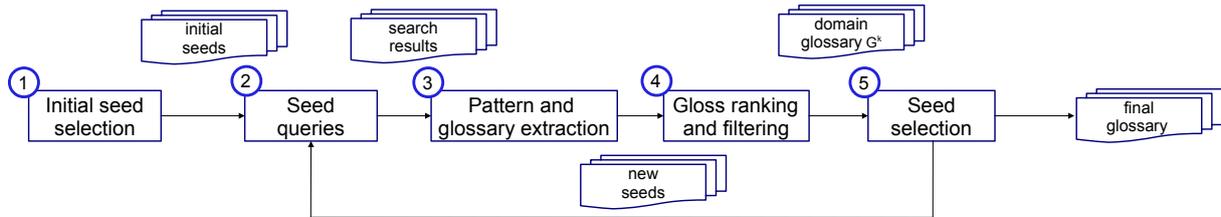


Figure 1: The GlossBoot bootstrapping process for glossary learning.

full-fledged domain glossary. Given a domain and a language of interest, we bootstrap the glossary learning process with just a few hypernymy relations (such as *computer is-a device*), with the only condition that the (term, hypernym) pairs must be specific enough to implicitly identify the domain in the target language. Hence we drop the requirement of a large domain corpus, and also avoid the use of training data or a manually defined set of lexical patterns. To the best of our knowledge, this is the first approach which jointly acquires large amounts of terms and glosses from the Web with minimal supervision for any target domain and language.

## 2 GlossBoot

Our objective is to harvest a domain glossary  $G$  containing pairs of terms/glosses in a given language. To this end, we automatically populate a set of HTML patterns  $P$  which we use to extract definitions from Web glossaries. Initially, both  $P := \emptyset$  and  $G := \emptyset$ . We incrementally populate the two sets by means of an initial seed selection step and four iterative steps (cf. Figure 1):

**Step 1. Initial seed selection:** first, we manually select a set of  $K$  hypernymy relation seeds  $S = \{(t_1, h_1), \dots, (t_K, h_K)\}$ , where the pair  $(t_i, h_i)$  contains a term  $t_i$  and its generalization  $h_i$  (e.g., (*firewall*, *security system*)). This is the only human input to the entire glossary learning process. The selection of the input seeds plays a key role in the bootstrapping process, in that the pattern and gloss extraction process will be driven by these seeds. The chosen hypernymy relations thus have to be as topical and representative as possible for the domain of interest (e.g., (*compiler*, *computer program*) is an appropriate pair for computer science, while (*byte*, *unit of measurement*) is not, as it might cause the extraction of several glossaries of units and measures).

We now set the iteration counter  $k$  to 1 and start the first iteration of the glossary bootstrapping pro-

cess (steps 2-5). After each iteration  $k$ , we keep track of the set of glosses  $G^k$ , acquired during iteration  $k$ .

**Step 2. Seed queries:** for each seed pair  $(t_i, h_i)$ , we submit the following query to a Web search engine: “ $t_i$ ” “ $h_i$ ” `glossaryKeyword`<sup>2</sup> (where `glossaryKeyword` is the term in the target language referring to *glossary* (i.e., *glossary* for English, *glossaire* for French etc.)) and collect the top-ranking results for each query.<sup>3</sup> Each resulting page is a candidate glossary for the domain implicitly identified by our relation seeds  $S$ .

**Step 3. Pattern and glossary extraction:** we initialize the glossary for iteration  $k$  as follows:  $G^k := \emptyset$ . Next, from each resulting page, we harvest all the text snippets  $s$  starting with  $t_i$  and ending with  $h_i$  (e.g., “*firewall*</b> – a <i>*security system*” where  $t_i = \textit{firewall}$  and  $h_i = \textit{security system}$ ), i.e.,  $s = t_i \dots h_i$ . For each such text snippet  $s$ , we perform the following substeps:

**(a) extraction of the term/gloss separator:** we start from  $t_i$  and move right until we extract the longest sequence  $p_M$  of HTML tags and non-alphanumeric characters, which we call the *term/gloss separator*, between  $t_i$  and the glossary definition (e.g., “</b> –” between “*firewall*” and “*a*” in the above example).

**(b) gloss extraction:** we expand the snippet  $s$  to the right of  $h_i$  in search of the entire gloss of  $t_i$ , i.e., until we reach a block element (e.g., <span>, <p>, <div>), while ignoring formatting elements such as <b>, <i> and <a> which are typically included within a definition sentence. As a result, we obtain the sequence  $t_i p_M gloss_s(t_i) p_R$ , where  $gloss_s(t_i)$  is our gloss for seed term  $t_i$  in snippet  $s$  (which includes  $h_i$  by construction) and  $p_R$  is the HTML block element

<sup>2</sup>In what follows we use the `typewriter` font for keywords and term/gloss separators.

<sup>3</sup>We use the Google Ajax APIs, which return the 64 top-ranking search results.

Generalized pattern	HTML text snippet
$\langle \text{strong} \rangle * \langle / \text{strong} \rangle - * \langle / \text{span} \rangle$	$\langle \text{strong} \rangle$ Interrupt $\langle / \text{strong} \rangle$ - The suspension of normal program execution to perform a higher priority service routine as requested by a peripheral device. $\langle / \text{span} \rangle$
$\langle \text{dt} \rangle * \langle / \text{dt} \rangle \langle \text{dd} \rangle * \langle / \text{dd} \rangle$	$\langle \text{dt} \rangle$ Netiquette $\langle / \text{dt} \rangle \langle \text{dd} \rangle$ The established conventions of online politeness are called netiquette. $\langle / \text{dd} \rangle$
$\langle \text{h3} \rangle * \langle / \text{h3} \rangle \langle \text{p} \rangle * \langle / \text{p} \rangle$	$\langle \text{h3} \rangle$ Compiler $\langle / \text{h3} \rangle \langle \text{p} \rangle$ A program that translates source code, such as C++ or Pascal, into directly executable machine code. $\langle / \text{p} \rangle$
$\langle \text{span} \rangle * \langle / \text{span} \rangle - * \langle / \text{p} \rangle$	$\langle \text{span} \rangle$ Signature $\langle / \text{span} \rangle$ - A function’s name and parameter list. $\langle / \text{p} \rangle$
$\langle \text{span} \rangle * \langle / \text{span} \rangle : * \langle \text{span} \rangle$	$\langle \text{span} \rangle$ Blog $\langle / \text{span} \rangle$ : Short for “web log”, a blog is an online journal. $\langle \text{span} \rangle$

Table 1: Examples of generalized patterns together with matching HTML text snippets.

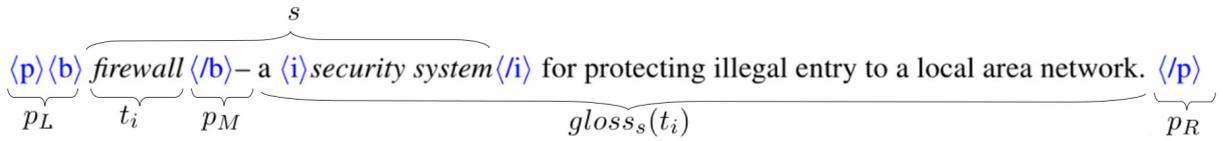


Figure 2: An example of decomposition during pattern extraction for a snippet matching the seed pair (*firewall*, *security system*).

to the right of the extracted gloss. In Figure 2 we show the decomposition of our example snippet matching the seed (*firewall*, *security system*).

**(c) pattern instance extraction:** we extract the following pattern instance:

$$p_L t_i p_M gloss_s(t_i) p_R,$$

where  $p_L$  is the longest sequence of HTML tags and non-alphanumeric characters obtained when moving to the left of  $t_i$  (see Figure 2).

**(d) pattern extraction:** we generalize the above pattern instance to the following pattern:

$$p_L * p_M * p_R,$$

i.e., we replace  $t_i$  and  $gloss_s(t_i)$  with  $*$ . For the above example, we obtain the following pattern:

$$\langle \text{p} \rangle \langle \text{b} \rangle * \langle / \text{b} \rangle - * \langle / \text{p} \rangle.$$

Finally, we add the generalized pattern to the set of patterns  $P$ , i.e.,  $P := P \cup \{p_L * p_M * p_R\}$ . We also add the first sentence of the retrieved gloss  $gloss_s(t_i)$  to our glossary  $G^k$ , i.e.,  $G^k := G^k \cup \{(t_i, first(gloss_s(t_i)))\}$ , where  $first(g)$  returns the first sentence of gloss  $g$ .

**(e) pattern matching:** finally, we look for additional pairs of terms/glosses in the Web page containing the snippet  $s$  by matching the page against the generalized pattern  $p_L * p_M * p_R$ . We then

add to  $G^k$  the new (term, gloss) pairs matching the generalized pattern. In Table 1 we show some non-trivial generalized patterns together with matching HTML text snippets.

As a result of step 3, we obtain a glossary  $G^k$  for the terms discovered at iteration  $k$ .

**Step 4. Gloss ranking and filtering:** importantly, not all the extracted definitions pertain to the domain of interest. In order to rank the glosses obtained at iteration  $k$  by domain pertinence, we assume that the terms acquired at previous iterations belong to the target domain, i.e., they are domain terms at iteration  $k$ . Formally, we define the terminology  $T_1^{k-1}$  of the domain terms accumulated up until iteration  $k - 1$  as follows:  $T_1^{k-1} := \bigcup_{i=1}^{k-1} T^i$ , where  $T^i := \{t : \exists(t, g) \in G^i\}$ . For the base step  $k = 1$ , we define  $T_1^0 := \{t : \exists(t, g) \in G^1\}$ , i.e., we use the first-iteration terminology itself.

To rank the glosses, we first transform each acquired gloss  $g$  to its bag-of-words representation  $Bag(g)$ , which contains all the single- and multi-word expressions in  $g$ . We use the lexicon of the target language’s Wikipedia together with  $T_1^{k-1}$  in order to obtain the bag of content words.<sup>4</sup> Then we

<sup>4</sup>In fact Wikipedia is only utilized in the multi-word identification phase. We do not use Wikipedia for discovering new terms.

Term	Gloss	Hypernym	# Seeds	Score
dynamic packet filter	A <b>firewall</b> <i>facility</i> that can monitor the <i>state</i> of active <u>connections</u> and use this <u>information</u> to determine which <u>network</u> packets to allow through the <b>firewall</b>	firewall	2	0.75
die	An integrated <i>circuit</i> <b>chip</b> <u>cut</u> from a finished <u>wafer</u> .	integrated circuit	1	0.75
constructor	a <u>method</u> used to help create a new <u>object</u> and initialise its <u>data</u>	method	0	1.00

Table 2: Examples of extracted terms, glosses and hypernyms (seeds are in bold, domain terms, i.e., in  $T_1^{k-1}$ , are underlined, non-domain terms in italics).

calculate the domain score of a gloss  $g$  as follows:

$$score(g) = \frac{|Bag(g) \cap T_1^{k-1}|}{|Bag(g)|}. \quad (1)$$

Finally, we use a threshold  $\theta$  (whose tuning is described in the experimental section) to remove from  $G^k$  those glosses  $g$  whose  $score(g) < \theta$ .

In Table 2 we show some glosses in the computer science domain (second column, domain terms are underlined) together with their scores (last column).

**Step 5. Seed selection for next iteration:** we now aim at selecting the new set of hypernymy relation seeds to be used to start the next iteration. We perform three substeps:

**(a) Hypernym extraction:** for each newly-acquired term/gloss pair  $(t, g) \in G^k$ , we automatically extract a candidate hypernym  $h$  from the textual gloss  $g$ . To do this we use a simple unsupervised heuristic which just selects the first term in the gloss.<sup>5</sup> We show an example of hypernym extraction for some terms in Table 2 (we report the term in column 1, the gloss in column 2 and the hypernyms extracted by the first term hypernym extraction heuristic in column 3).

**(b) (Term, Hypernym)-ranking:** we sort all the glosses in  $G^k$  by the number of seed terms found in each gloss. In the case of ties (i.e., glosses with the same number of seed terms), we further sort the glosses by the score given in Formula 1. We show an example of rank for some glosses in Table 2, where seed terms are in bold, domain terms (i.e., in  $T_1^{k-1}$ ) are underlined, and non-domain terms are shown in italics.

<sup>5</sup>While more complex strategies could be used, such as supervised classifiers (Navigli and Velardi, 2010), we found that this heuristic works well because, even when it is not a hypernym, the first term plays the role of a cue word for the defined term.

**(c) New seed selection:** we select the (term, hypernym) pairs corresponding to the  $K$  top-ranking glosses.

Finally, if  $k$  equals the maximum number of iterations, we stop. Else, we increment the iteration counter (i.e.,  $k := k + 1$ ) and jump to step (2) of our glossary bootstrapping algorithm after replacing  $S$  with the new set of seeds.

The output of glossary bootstrapping is a domain glossary  $G := \bigcup_{i=1, \dots, max} G^i$ , which includes a domain terminology  $T := \{t : \exists(t, g) \in G\}$  (i.e.,  $T := T_1^{max}$ ) and a set of glosses  $glosses(t)$  for each term  $t \in T$  (i.e.,  $glosses(t) := \{g : \exists(t, g) \in G\}$ ).

## 3 Experimental Setup

### 3.1 Domains and Gold Standards

For our experiments we focused on four different domains, namely, Computing, Botany, Environment, and Finance, and on three languages, namely, English, French and Italian. Note that not all the four domains are clear-cut. For instance, the Environment domain is quite interdisciplinary, including terms from fields such as Chemistry, Biology, Law, Politics, etc.

For each domain and language we selected as gold standards well-reputed glossaries on the Web, such as: the Utah computing glossary,<sup>6</sup> the Wikipedia glossary of botanical terms,<sup>7</sup> a set of Wikipedia glossaries about environment,<sup>8</sup> and the Reuters glossary for Finance<sup>9</sup> (full list at <http://lcl.uniroma1.it/glossboot/>). We report the size of the four gold-standard datasets in Table 4.

<sup>6</sup><http://www.math.utah.edu/~wisnia/glossary.html>

<sup>7</sup>[http://en.wikipedia.org/wiki/Glossary\\_of\\_botanical\\_terms](http://en.wikipedia.org/wiki/Glossary_of_botanical_terms)

<sup>8</sup>[http://en.wikipedia.org/wiki/List\\_of\\_environmental\\_issues](http://en.wikipedia.org/wiki/List_of_environmental_issues), [http://en.wikipedia.org/wiki/Glossary\\_of\\_environmental\\_science](http://en.wikipedia.org/wiki/Glossary_of_environmental_science), [http://en.wikipedia.org/wiki/Glossary\\_of\\_climate\\_change](http://en.wikipedia.org/wiki/Glossary_of_climate_change)

<sup>9</sup>[http://glossary.reuters.com/index.php/Main\\_Page](http://glossary.reuters.com/index.php/Main_Page)

Computing		Botany		Environment		Finance	
chip	circuit	leaf	organ	sewage	waste	eurobond	bond
destructor	method	grass	plant	acid rain	rain	asset play	stock
compiler	program	cultivar	variety	ecosystem	system	income stock	security
scanner	device	gymnosperm	plant	air monitoring	sampling	financial intermediary	institution
firewall	security system	flower	reproductive organ	global warming	temperature	derivative	financial product

Table 3: Hypernymy relation seeds used to bootstrap glossary learning in the four domains for the English language.

### 3.2 Seed Selection

For each domain and language we manually selected five seed hypernymy relations, shown for the English language in Table 3. The seeds were selected by the authors on the basis of just two conditions: i) the seeds should cover different aspects of the domain and should, indeed, identify the domain implicitly, ii) at least 10,000 results should be returned by the search engine when querying it with the seeds plus the `glossaryKeyword` (see step (2) of GlossBoot). The seed selection was not fine-tuned (i.e., it was not adjusted to improve performance), so it might well be that better seeds would provide better results (see, e.g., (Kozareva and Hovy, 2010b)). However, this type of consideration is beyond the scope of this paper.

#### 3.2.1 Evaluation measures

We performed experiments to evaluate the quality of both terms and glosses, as jointly extracted by GlossBoot.

**Terms.** For each domain and language we calculated coverage, extra-coverage and precision of the acquired terms  $T$ . Coverage is the ratio of extracted terms in  $T$  also contained in the gold standard  $\hat{T}$  to the size of  $\hat{T}$ . Extra-coverage is calculated as the ratio of the additional extracted terms in  $T \setminus \hat{T}$  over the number of gold standard terms  $\hat{T}$ . Finally, precision is the ratio of extracted terms in  $T$  deemed to be within the domain. To calculate precision we randomly sampled 5% of the retrieved terms and asked two human annotators to manually tag their domain pertinence (with adjudication in case of disagreement;  $\kappa = .62$ , indicating substantial agreement). Note that by sampling on the entire set  $T$ , we calculate the precision of both terms in  $T \cap \hat{T}$ , i.e., in the gold standard, and terms in  $T \setminus \hat{T}$ , i.e., not in the gold standard, which are not necessarily outside the domain.

**Glosses.** We calculated the precision of the extracted glosses as the ratio of glosses which were both well-formed textual definitions and specific

			Botany	Comput.	Environ.	Finance
EN	Gold std.	terms	772	421	713	1777
	GlossBoot	terms	5598	3738	4120	5294
		glosses	11663	4245	5127	6703
FR	Gold std.	terms	662	278	117	109
	GlossBoot	terms	3450	3462	1941	1486
		glosses	5649	3812	2095	1692
IT	Gold std.	terms	205	244	450	441
	GlossBoot	terms	1965	3356	1630	3601
		glosses	2678	5891	1759	5276

Table 4: Size of the gold-standard and automatically-acquired glossaries for the four domains in the three languages of interest.

to the target domain. Precision was determined on a random sample of 5% of the acquired glosses for each domain and language. The annotation was made by two annotators, with  $\kappa = .675$ , indicating substantial agreement.

### 3.3 Parameter tuning

We tuned the minimum and maximum length of both  $p_L$  and  $p_R$  (see step (3) of GlossBoot) and the threshold  $\theta$  that we use to filter out non-domain glosses (see step (4) of GlossBoot) using an extra domain, i.e., the Arts domain. To do this, we created a development dataset made up of the full set of 394 terms from the Tate Gallery glossary,<sup>10</sup> and bootstrapped our glossary extraction method with just one seed, i.e., (*fresco, painting*). We chose an optimal value of  $\theta = 0.1$  on the basis of a harmonic mean of coverage and precision. Note that, since precision also concerns terms not in the gold standard, we had to manually validate a sample of the extracted terms for each of the 21 tested values of  $\theta \in \{0, 0.05, 0.1, \dots, 1.0\}$ .

## 4 Results and Discussion

### 4.1 Terms

The size of the extracted terminologies for the four domains after five iterations are reported in Table 4. In Table 5 we show examples of the possible scenarios for terms: in-domain extracted terms

<sup>10</sup><http://www.tate.org.uk/collections/glossary/>

	In-domain (in gold std, $\in \hat{T} \cap T$ )	In-domain (not in gold std, $\in T \setminus \hat{T}$ )	Out-of-domain (not in gold std, $\in T \setminus \hat{T}$ )	In-domain (missed, $\in \hat{T} \setminus T$ )
Computing	software, inheritance, microprocessor	clipboard, even parity, sudoer	gs1-128 label, grayscale, quantum dots	openwindows, sun microsystems, hardwired
Botany	pollinium, stigma, spore	vegetation, dichogamous, fertilisation	ion, free radicals, mana-mana	nomenclature, endemism, insectivorous
Environment	carcinogen, footprint, solar power	frigid soil, biosafety, fire simulator	epidermis, science park, alum	g8, best practice, polystyrene
Finance	cash, bond, portfolio	truster, naked option, market price	precedent, immigration, heavy industry	co-location, petrodollars, euronext

Table 5: Examples of extracted (and missed) terms.

		Botany	Comput.	Environ.	Finance
EN	Precision	95%	98%	94%	98%
	Coverage	85%	40%	35%	32%
	Extra-coverage	640%	848%	542%	266%
FR	Precision	80%	97%	83%	98%
	Coverage	97%	27%	14%	26%
	Extra-coverage	425%	1219%	1646%	1350%
IT	Precision	89%	98%	76%	99%
	Coverage	42%	27%	11%	73%
	Extra-coverage	511%	1349%	356%	746%

Table 6: Precision, coverage and extra-coverage of the term extraction phase after 5 iterations.

which are also found in the gold standard (column 2), in-domain extracted terms but not in the gold standard (column 3), out-of-domain extracted terms (column 4), and domain terms in the gold standard but not extracted by our approach (column 5).

A quantitative evaluation is provided in Table 6, which shows the percentage results in terms of precision, coverage, and extra-coverage after 5 iterations of GlossBoot. For the English language we observe good coverage (between 32% and 40% on three domains, with a high peak of 85% coverage on Botany) and generally very high precision values. Moreover for the French and the Italian languages we observe a peak in the Botany and Finance domains respectively, while the lowest performances in terms of precision and coverage are observed for Environment, i.e., the most interdisciplinary domain.

In all three languages GlossBoot provides very high extra coverage of domain terms, i.e., additional terms which are not in the gold standard but are returned by our system. The figures, shown in Table 6, range between 266% (4726/1777) for the English Finance domain and 1646% (1926/117) for the French Environment domain. These results, together with the generally high precision values, indicate the larger extent of our bootstrapped glossaries compared to our gold standards.

Botany		Computing		Environm.		Finance	
Min	Max	Min	Max	Min	Max	Min	Max
26%	68%	8%	39%	5%	33%	14%	30%

Table 7: Coverage ranges for single-seed term extraction for the English language.

**Number of seeds.** Although the choice of selecting five hypernymy relation seeds is quite arbitrary, it shows that we can acquire a reliable terminology with minimal human intervention. Now, an obvious question arises: what if we bootstrapped GlossBoot with fewer hypernym seeds, e.g., just one seed? To answer this question we replicated our English experiments on each single (term, hypernym) pair in our seed set. In Table 7 we show the coverage ranges – i.e., the minimum and maximum coverage values – for the five seeds on each domain. We observe that the maximum coverage can attain values very close to those obtained with five seeds. However, the minimum coverage values are much lower. So, if we adopt a 1-seed bootstrapping policy there is a high risk of acquiring a poorer terminology unless we select the single seed very carefully, whereas we have shown that just a few seeds can cope with domain variability. Similar considerations can be made regarding different seed set sizes (we also tried 2, 3 and 4). So five is not a magic number, just one which can guarantee an adequate coverage of the domain.

**Number of iterations.** In order to study the coverage trend over iterations we selected 5 seeds for our tuning domain (i.e., Arts, see Section 3.3). Figure 3 shows the size (left graph), coverage, extra-coverage and precision (middle graph) of the acquired glossary after each iteration, from 1 to 20. As expected, (extra-)coverage grows over iterations, while precision drops. Stopping at iteration 5, as we do, is optimal in terms of the harmonic mean of precision and coverage (right graph in Figure 3).

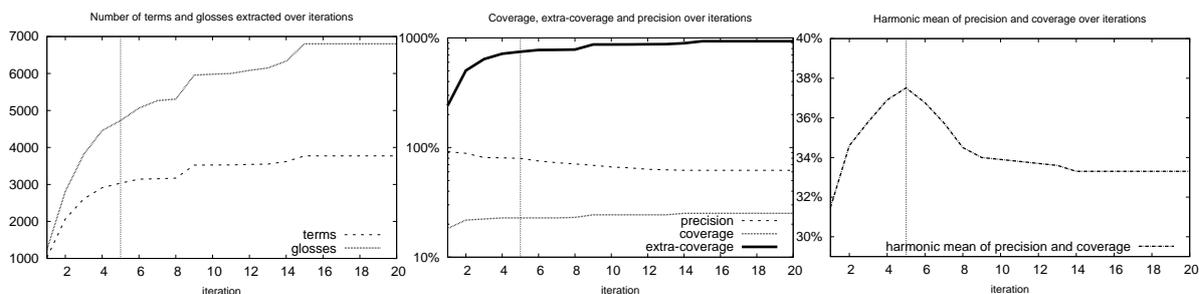


Figure 3: Size, coverage and precision trends for Arts (tuning domain) over 20 iterations for English.

	Botany	Comput.	Environm.	Finance
EN	96%	94%	97%	97%
FR	88%	89%	88%	95%
IT	94%	98%	83%	99%

Table 8: Precision of the glosses for the four domains and for the three languages.

## 4.2 Glosses

We show the results of gloss evaluation in Table 8. Precision ranges between 83% and 99%, with three domains performing above 92% on average across languages, and the Environment domain performing relatively worse because of its highly interdisciplinary nature (89% on average). We observe that these results are strongly correlated with the precision of the extracted terms (cf. Table 6), because the retrieved glosses of domain terms are usually in-domain too, and follow a definitional style because they come from glossaries. Note, however, that the gloss precision can also be higher than term precision, because many pertinent glosses might be extracted for the same term, cf. Table 4.

## 5 Comparative Evaluation

### 5.1 Comparison with Google Define

We performed a comparison with Google Define,<sup>11</sup> a state-of-the-art definition search service. This service inputs a term query and outputs a list of glosses. First, we randomly sampled 100 terms from our gold standard for each domain and each of the three languages. Next, for each domain and language, we manually calculated the fraction of terms for which an in-domain definition was provided by Google Define and GlossBoot. Table 9 shows the coverage results.

Google Define outperforms our system on all four domains (with a few exceptions). However

<sup>11</sup> Accessible from Google search by means of the `define: keyword`.

		Botany	Comput.	Environm.	Finance
EN	Google Define	90%	87%	84%	82%
	GlossBoot	77%	47%	44%	51%
FR	Google Define	40%	48%	36%	82%
	GlossBoot	88%	42%	22%	32%
IT	Google Define	52%	74%	78%	80%
	GlossBoot	64%	38%	44%	92%

Table 9: Number of domain glosses (from a random sample of 100 gold standard terms per domain) retrieved using Google Define and GlossBoot.

we note that Google Define: i) requires knowing the domain term to be defined in advance, whereas we jointly acquire thousands of terms and glosses starting from just a few seeds; ii) does not discriminate between glosses pertaining to the target domain and glosses concerning other fields or senses, whereas we extract domain-specific glosses.

### 5.2 Comparison with TaxoLearn

We also compared GlossBoot with a recent approach to glossary learning embedded into a framework for graph-based taxonomy learning from scratch, called TaxoLearn (Navigli et al., 2011). Since this approach requires the manual selection of a domain corpus to automatically extract terms and glosses, we decided to keep a level playing field and experimented with the same domain used by the authors, i.e., Artificial Intelligence (AI). TaxoLearn was applied to the entire set of IJCAI 2009 proceedings, resulting in the extraction of 427 terms and 834 glosses.<sup>12</sup> As regards GlossBoot, we selected 10 seeds to cover all the fields of AI, obtaining 5827 terms and 6716 glosses after 5 iterations, one order of magnitude greater than TaxoLearn.

As for the precision of the extracted terms, we randomly sampled 50% of them for each system. We show in Table 10 (first row) the estimated term

<sup>12</sup> Available at: <http://lcl.uniroma1.it/taxolearn>

	GlossBoot	TaxoLearn
Term Precision	82.3% (2398/2913)	77.0% (164/213)
Gloss Precision	82.8% (2780/3358)	78.9% (329/417)

Table 10: Estimated term and gloss precision of GlossBoot and TaxoLearn for the Artificial Intelligence domain.

precision for GlossBoot and TaxoLearn. The precision value for GlossBoot is lower than the precision values of the four domains in Table 6, due to the AI domain being highly interdisciplinary. TaxoLearn obtained a lower precision because it acquires a full-fledged taxonomy for the domain, thus also including higher-level concepts which do not necessarily pertain to the domain.

We performed a similar evaluation for the precision of the acquired glosses, by randomly sampling 50% of them for each system. We show in Table 10 (second row) the estimated gloss precision of GlossBoot and TaxoLearn. Again, GlossBoot outperforms TaxoLearn, retrieving a larger amount of glosses (6716 vs. 834) with higher precision. We remark, however, that in TaxoLearn glossary extraction is a by-product of the taxonomy learning process.

Finally, we note that we cannot compare with approaches based on lexical patterns (such as (Kozareva and Hovy, 2010a)), because they are not aimed at learning glossaries, but just at retrieving sentence snippets which contain pairs of terms/hypernyms (e.g., “*supervised systems* such as *decision trees*”).

## 6 Related Work

There are several techniques in the literature for the automated acquisition of definitional knowledge. Fujii and Ishikawa (2000) use an n-gram model to determine the definitional nature of text fragments, whereas Klavans and Muresan (2001) apply pattern matching techniques at the lexical level guided by cue phrases such as “is called” and “is defined as”. Cafarella et al. (2005) developed a Web search engine which handles more general and complex patterns like “*cities* such as *ProperNoun(Head(NP))*” in which it is possible to constrain the results with syntactic properties. More recently, a domain-independent supervised approach was presented which learns Word-Class Lattices (WCLs), i.e. lattice-based definition classifiers that are applied to candidate sentences containing the input terms (Navigli and Velardi, 2010). WCLs have been shown to perform with

high precision in several domains (Velardi et al., 2013).

To avoid the burden of manually creating a training dataset, definitional patterns can be extracted automatically. Reiplinger et al. (2012) experimented with two different approaches for the acquisition of lexical-syntactic patterns. The first approach involves bootstrapping patterns from a domain corpus, and then manually refining the acquired patterns. The second approach, instead, involves automatically acquiring definitional sentences by using a more sophisticated syntactic and semantic processing. The results shows high precision in both cases.

However, these approaches to glossary learning extract unrestricted textual definitions from open text. In order to filter out non-domain definitions, Velardi et al. (2008) automatically extract a domain terminology from an input corpus which they later use for assigning a domain score to each harvested definition and filtering out non-domain candidates. The extraction of domain terms from corpora can be performed either by means of statistical measures such as specificity and cohesion (Park et al., 2002), or just TF\*IDF (Kim et al., 2009).

To avoid the use of a large domain corpus, terminologies can be obtained from the Web by using Doubly-Anchored Patterns (DAPs) which, given a (term, hypernym) pair, harvest sentences matching manually-defined patterns like “<hypernym> such as <term>, and \*” (Kozareva et al., 2008). Kozareva and Hovy (2010a) further extend this term extraction process by harvesting new hypernyms using the corresponding inverse patterns (called  $DAP^{-1}$ ) like “\* such as <term<sub>1</sub>>, and <term<sub>2</sub>>”. Similarly to our approach, they drop the requirement of a domain corpus and start from a small number of (term, hypernym) seeds. However, while Doubly-Anchored Patterns have proven useful in the induction of domain taxonomies (Kozareva and Hovy, 2010a), they cannot be applied to the glossary learning task, because the extracted sentences are not formal definitions.

In contrast, GlossBoot performs the novel task of multilingual glossary learning from the Web by bootstrapping the extraction process with a few (term, hypernym) seeds. Bootstrapping techniques (Brin, 1998; Agichtein and Gravano, 2000; Paşca et al., 2006) have been successfully applied to several tasks, including high-precision semantic lexicon extraction from large corpora (Riloff and Jones, 1999; Thelen and Riloff, 2002; McIntosh

	Domain	Term	Gloss
EN	Botany	deciduous	losing foliage at the end of the growing season.
	Computing	information space	The abstract concept of everything accessible using networks: the Web.
	Finance	discount	The difference between the lower price paid for a security and the security's face amount at issue.
FR	Botany	insectivore	Qui capture des insectes et en absorbe les matières nutritives.
	Computing	notebook	C'est l'appellation d'un petit portable d'une taille proche d'une feuille A4.
	Environment	écosystème	Ensemble des êtres vivants et des éléments non vivants d'un milieu qui sont liés vitalement entre eux.
IT	Computing	link	Collegamento tra diverse pagine web, può essere costituito da immagini o testo.
	Environment	effetto serra	Riscaldamento dell'atmosfera terrestre dovuto alla presenza di gas nell'atmosfera (anidride carbonica, metano e vapore acqueo) che ostacolano l'uscita delle radiazioni infrarosse emesse dal suolo terrestre verso l'alto.
	Finance	spread	Indica la differenza tra la quotazione di acquisto e quella di vendita.

Table 11: An excerpt of the domain glossaries acquired for the three languages.

and Curran, 2008; McIntosh and Curran, 2009), learning semantic relations (Pantel and Pennacchiotti, 2006), extracting surface text patterns for open-domain question answering (Ravichandran and Hovy, 2002), semantic tagging (Huang and Riloff, 2010) and unsupervised Word Sense Disambiguation (Yarowsky, 1995). By exploiting the (term, hypernym) seeds to bootstrap the iterative acquisition of extraction patterns from Web glossary pages, we can cover the high variability of textual definitions, including both sentences matching the above-mentioned lexico-syntactic patterns (e.g., “a corpus is a collection of documents”) and glossary-style definitions (e.g., “corpus: a collection of document”) independently of the target domain and language.

## 7 Conclusions

In this paper we have presented GlossBoot, a new, minimally-supervised approach to multilingual glossary learning. Starting from a few hypernymy relation seeds which implicitly identify the domain of interest, we apply a bootstrapping approach which iteratively obtains HTML patterns from Web glossaries and then applies them to the extraction of term/gloss pairs. To our knowledge, GlossBoot is the first approach to large-scale glossary learning which jointly acquires thousands of terms and glosses for a target domain and language with minimal supervision.

The gist of GlossBoot is our glossary bootstrapping approach, thanks to which we can drop the requirements of existing techniques such as the availability of domain text corpora, which often do not contain enough definitions, and the man-

ual specification of lexical patterns, which typically extract sentence snippets, instead of formal glosses.

GlossBoot will be made available to the research community as open-source software. Beyond the immediate usability of its output and its effective use for domain Word Sense Disambiguation (Faralli and Navigli, 2012), we wish to show the benefit of GlossBoot in gloss-driven approaches to ontology learning (Navigli et al., 2011; Velardi et al., 2013) and semantic network enrichment (Navigli and Ponzetto, 2012). In Table 11 we show an excerpt of the acquired glossaries. All the glossaries and gold standards created for our experiments are available from the authors' Web site <http://lcl.uniroma1.it/glossboot/>.

We remark that the terminologies covered with GlossBoot are not only precise, but also one order of magnitude greater than those covered in individual online glossaries. As future work we plan to study the ability of GlossBoot to acquire domain glossaries at different levels of specificity (i.e., domains vs. subdomains). We also plan to exploit the acquired HTML patterns for implementing an open-source glossary crawler, along the lines of Google Define.

## Acknowledgments

The authors gratefully acknowledge the support of the ERC Starting Grant MultiJEDI No. 259234.



## References

- Eugene Agichtein and Luis Gravano. 2000. Snowball: extracting relations from large plain-text collections. In *Proceedings of the 5th ACM conference on Digital Libraries*, pages 85–94, San Antonio, Texas, USA.
- Sergey Brin. 1998. Extracting patterns and relations from the World Wide Web. In *Proceedings of the International Workshop on The World Wide Web and Databases*, pages 172–183, London, UK.
- Michael J. Cafarella, Doug Downey, Stephen Soderland, and Oren Etzioni. 2005. KnowItNow: Fast, scalable information extraction from the web. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 563–570, Vancouver, British Columbia, Canada.
- Hang Cui, Min-Yen Kan, and Tat-Seng Chua. 2007. Soft pattern matching models for definitional question answering. *ACM Transactions on Information Systems*, 25(2):1–30.
- Weisi Duan and Alexander Yates. 2010. Extracting glosses to disambiguate word senses. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 627–635, Los Angeles, CA, USA.
- Stefano Faralli and Roberto Navigli. 2012. A New Minimally-supervised Framework for Domain Word Sense Disambiguation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1411–1422, Jeju, Korea.
- Atsushi Fujii and Tetsuya Ishikawa. 2000. Utilizing the World Wide Web as an encyclopedia: extracting term descriptions from semi-structured texts. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 488–495, Hong Kong.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 539–545, Nantes, France.
- Eduard H. Hovy, Roberto Navigli, and Simone Paolo Ponzetto. 2013. Collaboratively built semi-structured content and Artificial Intelligence: The story so far. *Artificial Intelligence*, 194:2–27.
- Ruihong Huang and Ellen Riloff. 2010. Inducing domain-specific semantic class taggers from (almost) nothing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 275–285, Uppsala, Sweden.
- Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. 2009. An unsupervised approach to domain-specific term extraction. In *Proceedings of the Australasian Language Technology Workshop*, pages 94–98, Sydney, Australia.
- Judith Klavans and Smaranda Muresan. 2001. Evaluation of the DEFINDER system for fully automatic glossary construction. In *Proceedings of the American Medical Informatics Association (AMIA) Symposium*, pages 324–328, Washington, D.C., USA.
- Zornitsa Kozareva and Eduard Hovy. 2010a. A semi-supervised method to learn and construct taxonomies using the Web. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 1110–1118, Cambridge, MA, USA.
- Zornitsa Kozareva and Eduard H. Hovy. 2010b. Not all seeds are equal: Measuring the quality of text mining seeds. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 618–626, Los Angeles, California, USA.
- Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the Web with hyponym pattern linkage graphs. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 1048–1056, Columbus, Ohio, USA.
- Tara McIntosh and James R. Curran. 2008. Weighted mutual exclusion bootstrapping for domain independent lexicon and template acquisition. In *Proceedings of the Australasian Language Technology Association Workshop*, pages 97–105, CSIRO ICT Centre, Tasmania.
- Tara McIntosh and James R. Curran. 2009. Reducing semantic drift with bagging and distributional similarity. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 396–404, Suntec, Singapore.
- Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- Roberto Navigli and Paola Velardi. 2010. Learning Word-Class Lattices for definition and hypernym extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1318–1327, Uppsala, Sweden.
- Roberto Navigli, Paola Velardi, and Stefano Faralli. 2011. A graph-based algorithm for inducing lexical taxonomies from scratch. In *Proceedings of the 22th International Joint Conference on Artificial Intelligence*, pages 1872–1877, Barcelona, Spain.

- Marius Paşca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. 2006. Names and similarities on the web: Fact extraction in the fast lane. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 809–816, Sydney, Australia.
- Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, Sydney, Australia, pages 113–120, Sydney, Australia.
- Youngja Park, Roy J. Byrd, and Branimir K. Boguraev. 2002. Automatic glossary extraction: beyond terminology identification. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan.
- Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47, Philadelphia, Pennsylvania.
- Melanie Reiplinger, Ulrich Schäfer, and Magdalena Wolska. 2012. Extracting glossary sentences from scholarly articles: A comparative evaluation of pattern bootstrapping and deep analysis. In *Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries*, pages 55–65, Jeju Island, Korea.
- Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference*, pages 474–479, Menlo Park, CA, USA.
- Horacio Saggion. 2004. Identifying definitions in text collections for question answering. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, pages 1927–1930, Lisbon, Portugal.
- Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 214–221, Salt Lake City, UT, USA.
- Paola Velardi, Roberto Navigli, and Pierluigi D’Amadio. 2008. Mining the Web to create specialized glossaries. *IEEE Intelligent Systems*, 23(5):18–25.
- Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. OntoLearn Reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3).
- David Yarowsky. 1995. Unsupervised Word Sense Disambiguation rivaling supervised methods. In *Proceedings of the 33<sup>rd</sup> Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, MA, USA.

# Collective Annotation of Linguistic Resources: Basic Principles and a Formal Model

Ulle Endriss and Raquel Fernández

Institute for Logic, Language & Computation

University of Amsterdam

{ulle.endriss|raquel.fernandez}@uva.nl

## Abstract

Crowdsourcing, which offers new ways of cheaply and quickly gathering large amounts of information contributed by volunteers online, has revolutionised the collection of labelled data. Yet, to create annotated linguistic resources from this data, we face the challenge of having to combine the judgements of a potentially large group of annotators. In this paper we investigate how to aggregate individual annotations into a single collective annotation, taking inspiration from the field of social choice theory. We formulate a general formal model for collective annotation and propose several aggregation methods that go beyond the commonly used majority rule. We test some of our methods on data from a crowdsourcing experiment on textual entailment annotation.

## 1 Introduction

In recent years, the possibility to undertake large-scale annotation projects with hundreds or thousands of annotators has become a reality thanks to online crowdsourcing methods such as Amazon's *Mechanical Turk* and *Games with a Purpose*. Although these techniques open the door to a true revolution for the creation of annotated corpora, within the computational linguistics community there so far is no clear understanding of how the so-called "wisdom of the crowds" could or should be used to develop useful annotated linguistic resources. Those who have looked into this increasingly important issue have mostly concentrated on validating the quality of multiple non-expert annotations in terms of how they compare to expert gold standards; but they have only used simple aggregation methods based on majority voting to combine the judgments of individual annotators (Snow et al., 2008; Venhuizen et al., 2013).

In this paper, we take a different perspective and instead focus on investigating different aggregation methods for deriving a single *collective annotation* from a diverse set of judgments. For this we draw inspiration from the field of *social choice theory*, a theoretical framework for combining the preferences or choices of several individuals into a collective decision (Arrow et al., 2002). Our aim is to explore the parallels between the task of aggregating the preferences of the citizens participating in an election and the task of combining the expertise of speakers taking part in an annotation project. Our contribution consists in the formulation of a general formal model for collective annotation and, in particular, the introduction of several families of aggregation methods that go beyond the commonly used majority rule.

The remainder of this paper is organised as follows. In Section 2 we introduce some basic terminology and argue that there are four natural forms of collective annotation. We then focus on one of them and present a formal model for it in Section 3. We also formulate some basic principles of aggregation within this model in the same section. Section 4 introduces three families of aggregation methods: bias-correcting majority rules, greedy methods for identifying (near-)consensual coalitions of annotators, and distance-based aggregators. We test the former two families of aggregators, as well as the simple majority rule commonly used in similar studies, in a case study on data extracted from a crowdsourcing experiment on textual entailment in Section 5. Section 6 discusses related work and Section 7 concludes.

## 2 Four Types of Collective Annotation

An annotation task consists of a set of *items*, each of which is associated with a set of possible *categories* (Artstein and Poesio, 2008). The categories may be the same for all items or they may be item-specific. For instance, dialogue act annotation

(Allen and Core, 1997; Carletta et al., 1997) and word similarity rating (Miller and Charles, 1991; Finkelstein et al., 2002) involve choosing from amongst a set of categories—acts in a dialogue act taxonomy or values on a scale, respectively—which remains fixed for all items in the annotation task. In contrast, in tasks such as word sense labelling (Kilgarriff and Palmer, 2000; Palmer et al., 2007; Venhuizen et al., 2013) and PP-attachment annotation (Rosenthal et al., 2010; Jha et al., 2010) coders need to choose a category amongst a set of options specific to each item—the possible senses of each word or the possible attachment points in each sentence with a prepositional phrase.

In either case (one set of categories for all items vs. item-specific sets of categories), annotators are typically asked to identify, for each item, the category they consider the best match. In addition, they may be given the opportunity to indicate that they cannot judge (the “don’t know” or “unclear” category). For large-scale annotation projects run over the Internet it is furthermore very likely that an annotator will not be confronted with every single item, and it makes sense to distinguish items not seen by the annotator from items labelled as “don’t know”. We refer to this form of annotation, i.e., an annotation task where coders have the option to (i) label items with one of the available categories, to (ii) choose “don’t know”, or to (iii) not label an item at all, as *plain annotation*.

Plain annotation is the most common form of annotation and it is the one we shall focus on in this paper. However, other, more complex, forms of annotation are also possible and of interest. For instance, we may ask coders to *rank* the available categories (resulting in, say, a weak or partial order over the categories); we may ask them to provide a *qualitative ratings* of the available categories for each item (e.g., *excellent match*, *good match*, etc.); or we may ask for *quantitative ratings* (e.g., numbers from 1 to 100).<sup>1</sup> We refer to these forms of annotation as *complex annotation*.

We want to investigate how to aggregate the information available for each item once annotations by multiple annotators have been collected. In line with the terminology used in social choice theory and particularly judgment aggregation (Ar-

row, 1963; List and Pettit, 2002), let us call an aggregation method *independent* if the outcome regarding a given item  $j$  only depends on the categories provided by the annotators regarding  $j$  itself (but not on, say, the categories assigned to a different item  $j'$ ). Independent aggregation methods are attractive due to their simplicity. They also have some conceptual appeal: when deciding on  $j$  maybe we *should* only concern ourselves with what people have to say regarding  $j$ ? On the other hand, insisting on independence prevents us from exploiting potentially useful information that cuts across items. For instance, if a particular annotator almost always chooses category  $c$ , then we should maybe give less weight to her selecting  $c$  for the item  $j$  at hand than when some other annotator chooses  $c$  for  $j$ . This would call for methods that do not respect independence, which we shall refer to as *general* aggregation. Note that when studying independent aggregation methods, without loss of generality, we may assume that each annotation task consists of just a single item.

In view of our discussion above, there are four classes of approaches to collective annotation:

- (1) *Independent aggregation of plain annotations*. This is the simplest case, resulting in a fairly limited design space. When, for a given item, each annotator has to choose between  $k$  categories (or abstain) and we do not permit ourselves to use any other information, then the only reasonable choice is to implement the *plurality rule* (Taylor, 2005), under which the winning category is the category chosen by the largest number of annotators. In case there are exactly two categories available, the plurality rule is also called the *majority rule*. The only additional consideration to make here (besides how to deal with ties) is whether or not we may want to declare no winner at all in case the plurality winner does not win by a sufficiently significant margin or does not make a particular quota. This is the most common approach in the literature (see, e.g., Venhuizen et al., 2013).
- (2) *Independent aggregation of complex annotations*. This is a natural generalisation of the first approach, resulting in a wider range of possible methods. We shall not explore it here, but only point out that in case annotators provide *linear orders* over categories, there is a close resemblance to classical voting the-

<sup>1</sup>Some authors have combined qualitative and quantitative ratings; e.g., for the Graded Word Sense dataset of Erk et al. (2009) coders were asked to classify each relevant WordNet sense for a given item on a 5-point scale: 1 *completely different*, 2 *mostly different*, 3 *similar*, 4 *very similar*, 5 *identical*.

ory (Taylor, 2005); in case only *partial orders* can be elicited, recent work in computational social choice on the generalisation of classical voting rules may prove helpful (Pini et al., 2009; Endriss et al., 2009); and in case annotators rate categories using qualitative expressions such as *excellent match*, the method of *majority judgment* of Balinski and Laraki (2011) should be considered.

- (3) *General aggregation of plain annotations.* This is the approach we shall discuss below. It is related to *voting in combinatorial domains* studied in computational social choice (Chevaleyre et al., 2008), and to both *binary aggregation* (Dokow and Holzman, 2010; Grandi and Endriss, 2011) and *judgment aggregation* (List and Pettit, 2002).
- (4) *General aggregation of complex annotations.* While appealing due to its great level of generality, this approach can only be tackled successfully once approaches (2) and (3) are sufficiently well understood.

### 3 Formal Model

Next we present our model for general aggregation of plain annotations into a collective annotation.

#### 3.1 Terminology and Notation

An *annotation task* is defined in terms of  $m$  items, with each item  $j \in \{1, \dots, m\}$  being associated with a finite set of possible *categories*  $\mathcal{C}_j$ . Annotators are asked to provide an answer for each of the items of the annotation task. In the context of plain annotations, a valid *answer* for item  $j$  is an element of the set  $\mathcal{A}_j = \mathcal{C}_j \cup \{?, \perp\}$ .<sup>2</sup> Here  $?$  represents the answer “don’t know” and we use  $\perp$  to indicate that the annotator has not answered (or even seen) the item at all. An *annotation* is a vector of answers by one annotator, one answer for each item of the annotation task at hand, i.e., an annotation is an element of the Cartesian product  $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_m$ . A typical element of  $\mathcal{A}$  will be denoted as  $A = (a_1, \dots, a_m)$ .

Let  $\mathcal{N} = \{1, \dots, n\}$  be a finite set of  $n$  *annotators* (or *coders*). A *profile*  $\mathbf{A} = (A_1, \dots, A_n) \in \mathcal{A}^n$ , for a given annotation task, is a vector of annotations, one for each annotator. That is,  $\mathbf{A}$  is an

<sup>2</sup>As discussed earlier, in the context of *complex annotations*, an answer could also be, say, a partial order on  $\mathcal{C}_j$  or a function associating elements of  $\mathcal{C}_j$  with numerical ratings.

	Item 1	Item 2	Item 3
Annotator 1	B	A	A
Annotator 2	B	B	B
Annotator 3	A	B	A
Majority	B	B	A

Table 1: A profile with a collective annotation.

$n \times m$ -matrix; e.g.,  $a_{3,7}$  is the answer that the 3rd annotator provides for the 7th item.

We want to aggregate the information provided by the annotators into a (single) collective annotation. For the sake of simplicity, we use  $\mathcal{A}$  also as the domain of possible collective annotations (even though the distinction between  $?$  and  $\perp$  may not be strictly needed here; they both indicate that we do not want to commit to any particular category). An *aggregator* is a function  $F : \mathcal{A}^n \rightarrow \mathcal{A}$ , mapping any given profile into a collective annotation, i.e., a labelling of the items in the annotation task with corresponding categories (or  $?$  or  $\perp$ ). An example is the *plurality rule* (also known as the *majority rule* for binary tasks with  $|\mathcal{C}_j| = 2$  for all items  $j$ ), which annotates each item with the category chosen most often.

Note that the collective annotation need not coincide with any of the individual annotations. Take, for example, a binary annotation task in which three coders label three items with category A or B as shown in Table 1. Here using the majority rule to aggregate the annotations would result in a collective annotation that does not fully match any annotation by an individual coder.

#### 3.2 Basic Properties

A typical task in social choice theory is to formulate *axioms* that formalise specific desirable properties of an aggregator  $F$  (Arrow et al., 2002). Below we adapt three of the most basic axioms that have been considered in the social choice literature to our setting and we briefly discuss their relevance to collective annotation tasks.

We will require some additional notation: for any profile  $\mathbf{A}$ , item  $j$ , and possible answer  $a \in \mathcal{A}_j$ , let  $N_{j:a}^{\mathbf{A}}$  denote the set of annotators who chose answer  $a$  for item  $j$  under profile  $\mathbf{A}$ .

- $F$  is *anonymous* if it treats coders symmetrically, i.e., if for every permutation  $\pi : \mathcal{N} \rightarrow \mathcal{N}$ ,  $F(A_1, \dots, A_n) = F(A_{\pi(1)}, \dots, A_{\pi(n)})$ . In social choice theory, this is a fairness constraint. For us, fairness *per se* is not a desideratum,

but when we do not have any *a priori* information regarding the expertise of annotators, then anonymity is a natural axiom to adopt.

- $F$  is *neutral* if it treats all items symmetrically, i.e., if for every two items  $j$  and  $j'$  with the same set of possible categories (i.e., with  $C_j = C_{j'}$ ) and for every profile  $\mathbf{A}$ , it is the case that whenever  $N_{j:a}^{\mathbf{A}} = N_{j':a}^{\mathbf{A}}$  for all answers  $a \in \mathcal{A}_j = \mathcal{A}_{j'}$ , then  $F(\mathbf{A})_j = F(\mathbf{A})_{j'}$ . That is, if the patterns of individual annotations of  $j$  and  $j'$  are the same, then also their collective annotation should coincide. In social choice theory, neutrality is also considered a basic fairness requirement (avoiding preferential treatment one candidate in an election). In the context of collective annotation there may be good reasons to violate neutrality: e.g., we may use an aggregator that assigns different default categories to different items and that can override such a default decision only in the presence of a significant majority (note that this is different from anonymity: we will often not have any information on our annotators, but we may have tangible information on items).<sup>3</sup>
- $F$  is *independent* if the collective annotation of any given item  $j$  only depends on the individual annotations of  $j$ . Formally,  $F$  is independent if, for every item  $j$  and every two profiles  $\mathbf{A}$  and  $\mathbf{A}'$ , it is the case that whenever  $N_{j:a}^{\mathbf{A}} = N_{j:a}^{\mathbf{A}'}$  for all answers  $a \in \mathcal{A}_j$ , then  $F(\mathbf{A})_j = F(\mathbf{A}')_j$ . In social choice theory, independence is often seen as a desirable albeit hard (or even impossible) to achieve property (Arrow, 1963). For collective annotation, we strongly believe that it is *not* a desirable property: by considering how annotators label other items we can learn about their biases and we should try to exploit this information to obtain the best possible annotation for the item at hand.

Note that the plurality/majority rule is independent. All of the methods we shall propose in Section 4 are both anonymous and neutral—except to the extent to which we have to violate basic symmetry requirements in order to break ties between categories chosen equally often for a given item. None of our aggregators is independent.

<sup>3</sup>It would also be of interest to formulate a neutrality axiom w.r.t. categories (rather than items). For two categories, this idea has been discussed under the name of *domain-neutrality* in the literature (Grandi and Endriss, 2011), but for larger sets of categories it has not yet been explored.

Some annotation tasks might be subject to *integrity constraints* that determine the internal consistency of an annotation. For example, if our items are pairs of words and the possible categories include *synonymous* and *antonymous*, then if item 1 is about words  $A$  and  $B$ , item 2 about words  $B$  and  $C$ , and item 3 about words  $A$  and  $C$ , then any annotation that labels items 1 and 2 as *synonymous* should not label item 3 as *antonymous*. Thus, a further desirable property that will play a role for some annotation tasks is *collective rationality* (Grandi and Endriss, 2011): if all individual annotations respect a given integrity constraint, then so should the collective annotation.

We can think of integrity constraints as imposing top-down expert knowledge on an annotation. However, for some annotation tasks, no integrity constraints may be known to us in advance, even though we may have reasons to believe that the individual annotators do respect some such constraints. In that case, selecting one of the individual annotations in the profile as the collective annotation is the only way to ensure that these integrity constraints will be satisfied by the collective annotation (Grandi and Endriss, 2011). Of course, to do so we would need to assume that there is at least one annotator who has labelled *all* items (and to be able to design a high-quality aggregator in this way we should have a sufficiently large number of such annotators to choose from), which may not always be possible, particularly in the context of crowdsourcing.

## 4 Three Families of Aggregators

In this section we instantiate our formal model by proposing three families of methods for aggregation. Each of them is inspired, in part, by standard approaches to designing aggregation rules developed in social choice theory and, in part, by the specific needs of collective annotation. Regarding the latter point, we specifically emphasise the fact that not all annotators can be expected to be equally reliable (in general or w.r.t. certain items) and we try to integrate the process of aggregation with a process whereby less reliable annotators are either given less weight or are excluded altogether.

### 4.1 Bias-Correcting Majority Rules

We first want to explore the following idea: If a given annotator annotates *most* items with 0, then we might want to assign less significance to that

choice for any particular item.<sup>4</sup> That is, if an annotator appears to be biased towards a particular category, then we might want to try to correct for this bias during aggregation.

What follows applies only to annotation tasks where every item is associated with the same set of categories. For ease of exposition, let us furthermore assume that there are only two categories, 0 and 1, and that annotators do not make use of the option to annotate with ? (“don’t know”).

For every annotator  $i \in \mathcal{N}$  and every category  $X \in \{0, 1\}$ , fix a *weight*  $w_i^X \in \mathbb{R}$ . The *bias-correcting majority* (BCM) rule for this family of weights is defined as follows. Given profile  $\mathbf{A}$ , the collective category for item  $j$  will be 1 in case  $\sum_{a_{i,j}=1} w_i^1 > \sum_{a_{i,j}=0} w_i^0$ , and 0 otherwise.<sup>5</sup> That is, we compute the overall weight for category 1 by adding up the corresponding weights for those coders that chose 1 for item  $j$ , and we do accordingly for the overall weight for category 0; finally, we choose as collective category that category with the larger overall weight. Note that for  $w_i^X \equiv 1$  we obtain the simple majority rule.

Below we define three intuitively appealing families of weights, and thereby three BCM rules. However, before we do so, we first require some additional notation. Fix a profile of annotations. For  $X \in \{0, 1\}$ , let  $\text{Freq}_i(X)$  denote the relative frequency with which annotator  $i$  has chosen category  $X$ . For instance, if  $i$  has annotated 20 items and has chosen 1 in five cases, then  $\text{Freq}_i(1) = 0.25$ . Similarly, let  $\text{Freq}(X)$  denote the frequency of  $X$  across the entire profile.

Here are three ways of making the intuitive idea of bias correction concrete:

- (1) The *complement-based* BCM rule (ComBCM) is defined by weights  $w_i^X = \text{Freq}_i(1-X)$ . That is, the weight of annotator  $i$  for category  $X$  is equal to her relative frequency of having chosen the other category  $1-X$ . For example, if you annotate two items with 1 and eight with 0, then each of your 1-annotations will have weight 0.8, while each of your 0-annotations will only have weight 0.2.
- (2) The *difference-based* BCM rule (DiffBCM) is defined by weights  $w_i^X = 1 + \text{Freq}_i(X) -$

<sup>4</sup>A similar idea is at the heart of *cumulative voting*, which requires a voter to distribute a fixed number of points amongst the candidates (Glasser, 1959; Brams and Fishburn, 2002).

<sup>5</sup>For the sake of simplicity, our description here presupposes that ties are always broken in favour of 0. Other tie-breaking rules (e.g., random tie-breaking) are possible.

$\text{Freq}_i(X)$ . Recall that  $\text{Freq}(X)$  is the relative frequency of  $X$  in the entire profile, while  $\text{Freq}_i(X)$  is the relative frequency of  $X$  in the annotation of  $i$ . Hence, if  $i$  assigns category  $X$  less often than the general population, then her weight on  $X$ -choices will be increased by the difference (and *vice versa* in case she assigns  $X$  more often than the population at large). For example, if you assign 1 in two out of ten cases, while in general category 1 appears in exactly 50% of all annotations, then your weight for a choice of 1 will be  $1 + 0.5 - 0.2 = 1.3$ , while your weight for a choice of 0 will only be 0.7.

- (3) The *relative* BCM rule (RelBCM) is defined by weights  $w_i^X = \frac{\text{Freq}_i(X)}{\text{Freq}(X)}$ . The idea is very similar to the DiffBCM rule. For the example given above, your weight for a choice of 1 would be  $0.5/0.2 = 2.5$ , while your weight for a choice of 0 would be  $0.5/0.8 = 0.625$ .

The main difference between the ComBCM rule and the other two rules is that the former only takes into account the possible bias of individual annotators, while the latter two factor in as well the possible skewness of the data (as reflected by the labelling behaviour of the full set of annotators).

In addition, while ComBCM is specific to the case of two categories, DiffBCM and RelBCM immediately generalise to any number of categories. In this case, we add up the category-specific weights as before and then choose the category with maximal support (i.e., we generalise the majority rule underlying the family of BCM rules to the plurality rule).

We stress that our bias-correcting majority rules do *not* violate anonymity (nor neutrality for that matter). If we were to give less weight to a given annotator based on, say, her name, this would constitute a violation of anonymity; if we do so due to properties of the profile at hand and if we do so in a symmetric manner, then it does not.

## 4.2 Greedy Consensus Rules

Now consider the following idea: If for a given item there is almost complete consensus amongst those coders that annotated it with a proper category (i.e., those who did not choose ? or  $\perp$ ), then we should probably adopt their choice for the collective annotation. Indeed, most aggregators will make this recommendation. Furthermore, the fact that there is almost full consensus for one item

may cast doubts on the reliability of coders who disagree with this near-consensus choice and we might want to disregard their views not only w.r.t. that item but also as far as the annotation of other items is concerned. Next we propose a family of aggregators that implement this idea.

For simplicity, suppose that the only proper categories available are 0 and 1 and that annotators do not make use of ? (but it is easy to generalise to arbitrary numbers of categories and scenarios where different items are associated with different categories). Fix a *tolerance value*  $t \in \{0, \dots, m\}$ . The *greedy consensus rule* GreedyCR<sup>t</sup> works as follows. First, initialise the set  $\mathcal{N}^*$  with the full population of annotators  $\mathcal{N}$ . Then iterate the following two steps:

- (1) Find the item with the strongest majority for either 0 or 1 amongst coders in  $\mathcal{N}^*$  and lock in that value for the collective annotation.
- (2) Eliminate all coders from  $\mathcal{N}^*$  who disagree on more than  $t$  items with the values locked in for the collective annotation so far.

Repeat this process until the categories for all  $m$  items have been settled.<sup>6</sup> We may think of this as a “greedy” way of identifying a coalition  $\mathcal{N}^*$  with high inter-annotator agreement and then applying the majority rule to this coalition to obtain the collective annotation.

To be precise, the above is a description of an entire *family* of aggregators: Whenever there is more than one item with a majority of maximal strength, we could choose to lock in any one of them. Also, when there is a split majority between annotators in  $\mathcal{N}^*$  voting 0 and those voting 1, we have to use a tie-breaking rule to make a decision. Additional heuristics may be used to make these local decisions, or they may be left to chance.

Note that in case  $t = m$ , GreedyCR<sup>t</sup> is simply the majority rule (as no annotator will ever get eliminated). In case  $t = 0$ , we end up with a coalition of annotators that unanimously agree with all of the categories chosen for the collective annotation. However, this coalition of perfectly aligned

<sup>6</sup>There are some similarities to Tideman’s *Ranked Pairs* method for preference aggregation (Tideman, 1987), which works by fixing the relative rankings of pairs of alternatives in order of the strength of the supporting majorities. In preference aggregation (unlike here), the population of voters is *not* reduced in the process; instead, decisions against the majority are taken whenever this is necessary to guarantee the transitivity of the resulting collective preference order.

annotators need not be the largest such coalition (due to the greedy nature of our rule).

Note that greedy consensus rules, as defined here, are both anonymous and neutral. Specifically, it is important not to confuse possible skewness of the data with a violation of neutrality of the aggregator.

### 4.3 Distance-based Aggregation

Our third approach is based on the notion of *distance*. We first define a metric on choices to be able to say how distant two choices are. This induces an aggregator that, for a given profile, returns a collective choice that minimises the sum of distances to the individual choices in the profile.<sup>7</sup> This opens up a wide range of possibilities; we only sketch some of them here.

A natural choice is the *adjusted Hamming distance*  $H : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R}_{\geq 0}$ , which counts how many items two annotations differ on:

$$H(A, A') = \sum_{j=1}^m \delta(a_j, a'_j)$$

Here  $\delta$  is the *adjusted discrete distance* defined as  $\delta(x, y) = 0$  if  $x = y$  or  $x \in \{?, \perp\}$  or  $y \in \{?, \perp\}$ , and as  $\delta(x, y) = 1$  in all other cases.<sup>8</sup>

Once we have fixed a distance  $d$  on  $\mathcal{A}$  (such as  $H$ ), this induces an aggregator  $F_d$ :

$$F_d(\mathbf{A}) = \operatorname{argmin}_{A \in \mathcal{A}} \sum_{i=1}^n d(A, A_i)$$

To be precise,  $F_d$  is an *irresolute* aggregator that might return a *set* of best annotations with minimal distance to the profile.

Note that  $F_H$  is simply the plurality rule. This is so because every element of the Cartesian product is a possible annotation. In the presence of integrity constraints excluding some combinations, however, a distance-based rule allows for more sophisticated forms of aggregation (by choosing the optimal annotation w.r.t. all feasible annotations).

We may also try to restrict the computation of distances to a subset of “reliable” annotators. Consider the following idea: If a group of annotators is (fairly) reliable, then they should have a

<sup>7</sup>This idea has been used in voting (Kemeny, 1959), belief merging (Konieczny and Pino Pérez, 2002), and judgment aggregation (Miller and Osherson, 2009).

<sup>8</sup>This  $\delta$ , divided by  $m$ , is the same thing as what Artstein and Poesio (2008) call the agreement value  $\operatorname{agr}_j$  for item  $j$ .

(fairly) high inter-annotator agreement. By this reasoning, we should choose a group of annotators  $\text{ANN} \subseteq \mathcal{N}$  that maximises inter-annotator agreement in ANN and work with the aggregator  $\text{argmin}_{A \in \mathcal{A}} \sum_{i \in \text{ANN}} d(A, A_i)$ . But this is too simplistic: any singleton  $\text{ANN} = \{i\}$  will result in perfect agreement. That is, while we can easily maximise agreement, doing so in a naïve way means ignoring most of the information collected. In other words, we face the following dilemma:

- On the one hand, we should choose a *small* set ANN (i.e., select *few* annotators to base our collective annotation on), as that will allow us to increase the (average) reliability of the annotators taken into account.
- On the other hand, we should choose a *large* set ANN (i.e., select *many* annotators to base our collective annotation on), as that will increase the amount of information exploited.

One pragmatic approach is to fix a minimum quality threshold regarding one of the two dimensions and optimise in view of the other.<sup>9</sup>

## 5 A Case Study

In this section, we report on a case study in which we have tested our bias-correcting majority and greedy consensus rules.<sup>10</sup> We have used the dataset created by Snow et al. (2008) for the task of recognising textual entailment, originally proposed by Dagan et al. (2006) in the PASCAL Recognizing Textual Entailment (RTE) Challenge. RTE is a binary classification task consisting in judging whether the meaning of a piece of text (the so-called hypothesis) can be inferred from another piece of text (the entailing text). The original RTE1 Challenge testset consists of 800 text-hypothesis pairs (such as  $T$ : “*Chrétien visited Peugeot’s newly renovated car factory*”,  $H$ : “*Peugeot manufactures cars*”) with a gold standard annotation that classifies each item as either *true* (1)—in case  $H$  can be inferred from  $T$ —or *false* (0). Exactly 400 items are annotated as 0 and exactly 400 as 1. Bos and Markert (2006) performed an independent expert annotation of

<sup>9</sup>GreedyCR<sup>*t*</sup> is a greedy (rather than optimal) implementation of this basic idea, with the tolerance value  $t$  fixing a threshold on (a particular form of) inter-annotator agreement.

<sup>10</sup>Since the annotation task and dataset used for our case study do not involve any interesting integrity constraints, we have not tested any distance-based aggregation rules.

this testset, obtaining 95% agreement between the RTE1 gold standard and their own annotation.

The dataset of Snow et al. (2008) includes 10 non-expert annotations for each of the 800 items in the RTE1 testset, collected with Amazon’s *Mechanical Turk*. A quick examination of the dataset shows that there are a total of 164 annotators who have annotated between 20 items (124 annotators) and 800 items each (only one annotator). Non-expert annotations with category 1 (rather than 0) are slightly more frequent ( $\text{Freq}(1) \approx 0.57$ ).

We have applied our aggregators to this data and compared the outcomes with each other and to the gold standard. The results are summarised in Table 2 and discussed in the sequel. For each pair we report the *observed agreement*  $A_o$  (proportion of items on which two annotations agree) and, in brackets, Cohen’s *kappa*  $\kappa = \frac{A_o - A_e}{1 - A_e}$ , with  $A_e$  being the expected agreement for independent annotators (Cohen, 1960; Artstein and Poesio, 2008).

Note that there are several variants of the majority rule, depending on how we break ties. In Table 2,  $\text{Maj}^{1>0}$  is the majority rule that chooses 1 in case the number of annotators choosing 1 is equal to the number of annotators choosing 0 (and accordingly for  $\text{Maj}^{0>1}$ ). For 65 out of the 800 items there has been a tie (i.e., five annotators choose 0 and another five choose 1). This means that the tie-breaking rule used can have a significant impact on results. Snow et al. (2008) work with a majority rule where ties are broken uniformly at random and report an observed agreement (accuracy) between the majority rule and the gold standard of 89.7%. This is confirmed by our results: 89.7% is the mean of 87.5% (our result for  $\text{Maj}^{1>0}$ ) and 91.9% (our result for  $\text{Maj}^{0>1}$ ). If we break ties in the optimal way (in view of approximating the gold standard (which of course would not actually be possible without having access to that gold standard), then we obtain an observed agreement of 93.8%, but if we are unlucky and ties happen to get broken in the worst possible way, we obtain an observed agreement of only 85.6%.

For none of our bias-correcting majority rules did we encounter any ties. Hence, for these aggregators the somewhat arbitrary choices we have to make when breaking ties are of no significance, which is an important point in their favour. Observe that all of the bias-correcting majority rules approximate the gold standard better than the majority rule with uniformly random tie-breaking.

Annotation	Maj <sup>1&gt;0</sup>	Maj <sup>0&gt;1</sup>	ComBCM	DiffBCM	RelBCM	GreedyCR <sup>0</sup>	GreedyCR <sup>15</sup>
Gold Standard	87.5% (.75)	91.9% (.84)	91.1% (.80)	91.5% (.81)	90.8% (.80)	86.6% (.73)	92.5% (.85)
Maj <sup>1&gt;0</sup>		91.9% (.84)	88.9% (.76)	94.3% (.87)	94.0% (.87)	87.6% (.75)	91.5% (.83)
Maj <sup>0&gt;1</sup>			96.0% (.91)	97.6% (.95)	96.9% (.93)	89.0% (.78)	96.1% (.92)
ComBCM				94.6% (.86)		88.8% (.75)	93.9% (.86)
DiffBCM					98.8% (.97)	88.6% (.75)	94.8% (.88)
RelBCM						88.4% (.74)	93.8% (.86)
GreedyCR <sup>0</sup>							90.6% (.81)

Table 2: Observed agreement (and  $\kappa$ ) between collective annotations and the gold standard.

Recall that the greedy consensus rule is in fact a family of aggregators: whenever there is more than one item with a maximal majority, we may lock in any one of them. Furthermore, when there is a split majority, then ties may be broken either way. The results reported here refer to an implementation that always chooses the lexicographically first item amongst all those with a maximal majority and that breaks ties in favour of 1. These parameters yield neither the best or the worst approximations of the gold standard. We tested a range of tolerance values. As an example, Table 2 includes results for tolerance values 0 and 15. The coalition found for tolerance 0 consists of 46 annotators who all completely agree with the collective annotation; the coalition found for tolerance 15 consists of 156 annotators who all disagree with the collective annotation on at most 15 items. While GreedyCR<sup>0</sup> appears to perform rather poorly, GreedyCR<sup>15</sup> approximates the gold standard particularly well. This is surprising and suggests, on the one hand, that eliminating only the most extreme outlier annotators is a useful strategy, and on the other hand, that a high-quality collective annotation can be obtained from a group of annotators that disagree substantially.<sup>11</sup>

## 6 Related Work

There is an increasing number of projects using crowdsourcing methods for labelling data. Online *Games with a Purpose*, originally conceived by von Ahn and Dabbish (2004) to annotate images, have been used for a variety of linguistic tasks: Lafourcade (2007) created JeuxDeMots to develop a semantic network by asking players to label words with semantically related words; Phrase Detectives (Chamberlain et al., 2008) has been used to gather annotations on anaphoric coreference; and more recently Basile et al. (2012)

<sup>11</sup>Recall that 124 out of 164 coders only annotated 20 items each; a tolerance value of 15 thus is fairly lenient.

have developed the Wordrobe set of games for annotating named entities, word senses, homographs, and pronouns. Similarly, crowdsourcing via microworking sites like Amazon’s *Mechanical Turk* has been used in several annotation experiments related to tasks such as affect analysis, event annotation, sense definition and word sense disambiguation (Snow et al., 2008; Rumshisky, 2011; Rumshisky et al., 2012), amongst others.<sup>12</sup>

All these efforts face the problem of how to aggregate the information provided by a group of volunteers into a collective annotation. However, by and large, the emphasis so far has been on issues such as experiment design, data quality, and costs, with little attention being paid to the aggregation methods used, which are typically limited to some form of majority vote (or taking averages if the categories are numeric). In contrast, our focus has been on investigating different aggregation methods for arriving at a collective annotation.

Our work has connections with the literature on inter-annotator agreement. Agreement scores such as *kappa* are used to assess the quality of an annotation but do not play a direct role in constructing one single annotation from the labellings of several coders.<sup>13</sup> The methods we have proposed, in contrast, do precisely that. Still, agreement plays a prominent role in some of these methods. In our discussion of distance-based aggregation, we suggested how agreement can be used to select a subset of annotators whose individual annotations are minimally distant from the resulting collective annotation. Our greedy consensus rule also makes use of agreement to ensure a minimum level of consensus. In both cases, the aggregators have the effect of disregarding some outlier annotators.

<sup>12</sup>See also the papers presented at the NAACL 2010 Workshop on Creating Speech and Language Data with Amazon’s *Mechanical Turk* ([tinyurl.com/amtworkshop2010](http://tinyurl.com/amtworkshop2010)).

<sup>13</sup>Creating a gold standard often involves adjudication of disagreements by experts, or even the removal of cases with disagreement from the dataset. See, e.g., the papers cited by Beigman Klebanov and Beigman (2009).

Other researchers have explored ways to directly identify “low-quality” annotators. For instance, Snow et al. (2008) and Raykar et al. (2010) propose Bayesian methods for identifying and correcting annotators’ biases, while Ipeirotis et al. (2010) propose an algorithm for assigning a quality score to annotators that distinguishes intrinsic error rate from an annotator’s bias. In our approach, we do not directly rate annotators or recalibrate their annotations—rather, some outlier annotators get to play a marginal role in the resulting collective annotation as a side effect of the aggregation methods themselves.

Although in our case study we have tested our aggregators by comparing their outcomes to a gold standard, our approach to collective annotation itself does *not* assume that there is in fact a ground truth. Instead, we view collective annotations as reflecting the views of a community of speakers.<sup>14</sup> This contrasts significantly with, for instance, the machine learning literature, where there is a focus on estimating *the hidden true label* from a set of noisy labels using maximum-likelihood estimators (Dawid and Skene, 1979; Smyth et al., 1995; Raykar et al., 2010).

In application domains where it is reasonable to assume the existence of a ground truth and where we are able to model the manner in which individual judgments are being distorted relative to this ground truth, social choice theory provides tools (using again maximum-likelihood estimators) for the design of aggregators that maximise chances of recovering the ground truth for a given model of distortion (Young, 1995; Conitzer and Sandholm, 2005). In recent work, Mao et al. (2013) have discussed the use of these methods in the context of crowdsourcing. Specifically, they have designed an experiment in which the ground truth is defined unambiguously and known to the experiment designer, so as to be able to extract realistic models of distortion from the data collected in a crowdsourcing exercise.

## 7 Conclusions

We have presented a framework for combining the expertise of speakers taking part in large-scale

---

<sup>14</sup>In some domains, such as medical diagnosis, it makes perfect sense to assume that there is a ground truth. However, in tasks related to linguistic knowledge and language use such an assumption seems far less justified. Hence, a collective annotation may be the closest we can get to a representation of the linguistic knowledge/use of a linguistic community.

annotation projects. Such projects are becoming more and more common, due to the availability of online crowdsourcing methods for data annotation. Our work is novel in several respects. We have drawn inspiration from the field of social choice theory to formulate a general formal model for aggregation problems, which we believe sheds light on the kind of issues that arise when trying to build annotated linguistic resources from a potentially large group of annotators; and we have proposed several families of concrete methods for aggregating individual annotations that are more fine-grained than the standard majority rule that so far has been used across the board. We have tested some of our methods on a gold standard testset for the task of recognising textual entailment.

Our aim has been conceptual, namely to point out that it is important for computational linguists to reflect on the methods used when aggregating annotation information. We believe that social choice theory offers an appropriate general methodology for supporting this reflection. Importantly, this does not mean that the concrete aggregation methods developed in social choice theory are immediately applicable or that all the axioms typically studied in social choice theory are necessarily relevant to aggregating linguistic annotations. Rather, what we claim is that it is the *methodology* of social choice theory which is useful: to formally state desirable properties of aggregators as axioms and then to investigate which specific aggregators satisfy them. To put it differently: at the moment, researchers in computational linguistics simply use some given aggregation methods (almost always the majority rule) and judge their quality on how they fare in specific experiments—but there is no principled reflection on the methods themselves. We believe that this should change and hope that the framework outlined here can provide a suitable starting point.

In future work, the framework we have presented here should be tested more extensively, not only against a gold standard but also in terms of the usefulness of the derived collective annotations for training supervised learning systems. On the theoretical side, it would be interesting to study the axiomatic properties of the methods of aggregation we have proposed here in more depth and to define axiomatic properties of aggregators that are specifically tailored to the task of collective annotation of linguistic resources.

## References

- James Allen and Mark Core, 1997. *DAMSL: Dialogue Act Markup in Several Layers*. Discourse Resource Initiative.
- Kenneth J. Arrow, Armatya K. Sen, and Kotaro Suzumura, editors. 2002. *Handbook of Social Choice and Welfare*. North-Holland.
- Kenneth J. Arrow. 1963. *Social Choice and Individual Values*. John Wiley and Sons, 2nd edition. First edition published in 1951.
- Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.
- Michel Balinski and Rida Laraki. 2011. *Majority Judgment: Measuring, Ranking, and Electing*. MIT Press.
- Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. A platform for collaborative semantic annotation. In *Proc. 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2012)*, pages 92–96.
- Beata Beigman Klebanov and Eyal Beigman. 2009. From annotator agreement to noise models. *Computational Linguistics*, 35(4):495–503.
- Johan Bos and Katja Markert. 2006. Recognising textual entailment with robust logical inference. In *Machine Learning Challenges*, volume 3944 of *LNCS*, pages 404–426. Springer-Verlag.
- Steven J. Brams and Peter C. Fishburn. 2002. Voting procedures. In Kenneth J. Arrow, Armatya K. Sen, and Kotaro Suzumura, editors, *Handbook of Social Choice and Welfare*. North-Holland.
- Jean Carletta, Stephen Isard, Anne H. Anderson, Gwyneth Doherty-Sneddon, Amy Isard, and Jacqueline C. Kowtko. 1997. The reliability of a dialogue structure coding scheme. *Computational Linguistics*, 23:13–31.
- Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. 2008. Addressing the resource bottleneck to create large-scale annotated texts. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, volume 1 of *Research in Computational Semantics*, pages 375–380. College Publications.
- Yann Chevaleyre, Ulle Endriss, Jérôme Lang, and Nicolas Maudet. 2008. Preference handling in combinatorial domains: From AI to social choice. *AI Magazine*, 29(4):37–46.
- Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46.
- Vincent Conitzer and Tuomas Sandholm. 2005. Common voting rules as maximum likelihood estimators. In *Proc. 21st Conference on Uncertainty in Artificial Intelligence (UAI-2005)*.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges*, volume 3944 of *LNCS*, pages 177–190. Springer-Verlag.
- Alexander Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28.
- Elad Dokow and Ron Holzman. 2010. Aggregation of binary evaluations. *Journal of Economic Theory*, 145(2):495–511.
- Ulle Endriss, Maria Silvia Pini, Francesca Rossi, and K. Brent Venable. 2009. Preference aggregation over restricted ballot languages: Sincerity and strategy-proofness. In *Proc. 21st International Joint Conference on Artificial Intelligence (IJCAI-2009)*.
- Katrin Erk, Diana McCarthy, and Nicholas Gaylord. 2009. Investigations on word senses and word usages. In *Proc. 47th Annual Meeting of the Association for Computational Linguistics (ACL-2009)*, pages 10–18.
- Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppín. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.
- Gerald J. Glasser. 1959. Game theory and cumulative voting for corporate directors. *Management Science*, 5(2):151–156.
- Umberto Grandi and Ulle Endriss. 2011. Binary aggregation with integrity constraints. In *Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI-2011)*.
- Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality Management on Amazon Mechanical Turk. In *Proc. 2nd Human Computation Workshop (HCOMP-2010)*.
- Mukund Jha, Jacob Andreas, Kapil Thadani, Sara Rosenthal, and Kathleen McKeown. 2010. Corpus creation for new genres: A crowdsourced approach to PP attachment. In *Proc. NAACL-HLT Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk*, pages 13–20.
- John Kemeny. 1959. Mathematics without numbers. *Daedalus*, 88:577–591.
- Adam Kilgarriff and Martha Palmer. 2000. Introduction to the special issue on senseval. *Computers and the Humanities*, 34(1):1–13.
- Sébastien Konieczny and Ramón Pino Pérez. 2002. Merging information under constraints: A logical framework. *Journal of Logic and Computation*, 12(5):773–808.

- Mathieu Lafourcade. 2007. Making people play for lexical acquisition with the JeuxDeMots prototype. In *Proc. 7th International Symposium on Natural Language Processing*.
- Christian List and Philip Pettit. 2002. Aggregating sets of judgments: An impossibility result. *Economics and Philosophy*, 18(1):89–110.
- Andrew Mao, Ariel D. Procaccia, and Yiling Chen. 2013. Better human computation through principled voting. In *Proc. 27th AAAI Conference on Artificial Intelligence*.
- George A. Miller and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.
- Michael K. Miller and Daniel Osherson. 2009. Methods for distance-based judgment aggregation. *Social Choice and Welfare*, 32(4):575–601.
- Martha Palmer, Hoa Trang Dang, and Christiane Fellbaum. 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13(2):137–163.
- Maria Silvia Pini, Francesca Rossi, K. Brent Venable, and Toby Walsh. 2009. Aggregating partially ordered preferences. *Journal of Logic and Computation*, 19(3):475–502.
- Vikas Raykar, Shipeng Yu, Linda Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *The Journal of Machine Learning Research*, 11:1297–1322.
- Sara Rosenthal, William Lipovsky, Kathleen McKeown, Kapil Thadani, and Jacob Andreas. 2010. Towards semi-automated annotation for prepositional phrase attachment. In *Proc. 7th International Conference on Language Resources and Evaluation (LREC-2010)*.
- Anna Rumshisky, Nick Botchan, Sophie Kushkuley, and James Pustejovsky. 2012. Word sense inventories by non-experts. In *Proc. 8th International Conference on Language Resources and Evaluation (LREC-2012)*.
- Anna Rumshisky. 2011. Crowdsourcing word sense definition. In *Proc. ACL-HLT 5th Linguistic Annotation Workshop (LAW-V)*.
- Padhraic Smyth, Usama Fayyad, Michael Burl, Pietro Perona, and Pierre Baldi. 1995. Inferring ground truth from subjective labelling of venus images. *Advances in Neural Information Processing Systems*, pages 1085–1092.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP-2008)*, pages 254–263.
- Alan D. Taylor. 2005. *Social Choice and the Mathematics of Manipulation*. Cambridge University Press.
- T. Nicolaus Tideman. 1987. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206.
- Noortje Venhuizen, Valerio Basile, Kilian Evang, and Johan Bos. 2013. Gamification for word sense labeling. In *Proc. 10th International Conference on Computational Semantics (IWCS-2013)*, pages 397–403.
- Luis von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, pages 319–326. ACM.
- H. Peyton Young. 1995. Optimal voting rules. *Journal of Economic Perspectives*, 9(1):51–64.

# ParGramBank: The ParGram Parallel Treebank

**Sebastian Sulger and Miriam Butt**

University of Konstanz, Germany

{sebastian.sulger|miriam.butt}@uni-konstanz.de

**Tracy Holloway King**

eBay Inc., USA

tracyking@ebay.com

**Paul Meurer**

Uni Research AS, Norway

paul.meurer@uni.no

**Tibor Laczkó and György Rákosi**

University of Debrecen, Hungary

{laczko.tibor|rakosi.gyorgy}@arts.unideb.hu

**Cheikh Bamba Dione and Helge Dyvik and Victoria Rosén and Koenraad De Smedt**

University of Bergen, Norway

dione.bamba@lle.uib.no, {dyvik|victoria|desmedt}@uib.no

**Agnieszka Patejuk**

Polish Academy of Sciences

aep@ipipan.waw.pl

**Özlem Çetinoğlu**

University of Stuttgart, Germany

ozlem@ims.uni-stuttgart.de

**I Wayan Arka\*** and **Meladel Mistica<sup>+</sup>**

\*Australian National University and Udayana University, Indonesia

<sup>+</sup>Australian National University

wayan.arka@anu.edu.au, meladel.mistica@gmail.com

## Abstract

This paper discusses the construction of a parallel treebank currently involving ten languages from six language families. The treebank is based on deep LFG (Lexical-Functional Grammar) grammars that were developed within the framework of the ParGram (Parallel Grammar) effort. The grammars produce output that is maximally parallelized across languages and language families. This output forms the basis of a parallel treebank covering a diverse set of phenomena. The treebank is publicly available via the INESS treebanking environment, which also allows for the alignment of language pairs. We thus present a unique, multilayered parallel treebank that represents more and different types of languages than are available in other treebanks, that represents

deep linguistic knowledge and that allows for the alignment of sentences at several levels: dependency structures, constituency structures and POS information.

## 1 Introduction

This paper discusses the construction of a parallel treebank currently involving ten languages that represent several different language families, including non-Indo-European. The treebank is based on the output of individual deep LFG (Lexical-Functional Grammar) grammars that were developed independently at different sites but within the overall framework of ParGram (the Parallel Grammar project) (Butt et al., 1999a; Butt et al., 2002). The aim of ParGram is to produce deep, wide coverage grammars for a variety of languages. Deep grammars provide detailed syntactic analysis, encode grammatical functions as well as

other grammatical features such as tense or aspect, and are linguistically well-motivated. The ParGram grammars are couched within the linguistic framework of LFG (Bresnan, 2001; Dalrymple, 2001) and are constructed with a set of grammatical features that have been commonly agreed upon within the ParGram group. ParGram grammars are implemented using XLE, an efficient, industrial-strength grammar development platform that includes a parser, a generator and a transfer system (Crouch et al., 2012). XLE has been developed in close collaboration with the ParGram project. Over the years, ParGram has continuously grown and includes grammars for Arabic, Chinese, English, French, German, Georgian, Hungarian, Indonesian, Irish, Japanese, Malagasy, Murrinh-Patha, Norwegian, Polish, Spanish, Tigrinya, Turkish, Urdu, Welsh and Wolof.

ParGram grammars produce output that has been parallelized maximally across languages according to a set of commonly agreed upon universal proto-type analyses and feature values. This output forms the basis of the ParGramBank parallel treebank discussed here. ParGramBank is constructed using an innovative alignment methodology developed in the XPAR project (Dyvik et al., 2009) in which grammar parallelism is presupposed to propagate alignment across different projections (section 6). This methodology has been implemented with a drag-and-drop interface as part of the LFG Parsebanker in the INESS infrastructure (Rosén et al., 2012; Rosén et al., 2009). ParGramBank has been constructed in INESS and is accessible in this infrastructure, which also offers powerful search and visualization.

In recent years, parallel treebanking<sup>1</sup> has gained in importance within NLP. An obvious application for parallel treebanking is machine translation, where treebank size is a deciding factor for whether a particular treebank can support a particular kind of research project. When conducting in-depth linguistic studies of typological features, other factors such as the number of included languages, the number of covered phenomena, and the depth of linguistic analysis become more important. The treebanking effort reported on in this paper supports work of the latter focus, including efforts at multilingual dependency parsing (Naseem et al., 2012). We have

<sup>1</sup>Throughout this paper ‘treebank’ refers to both phrase-structure resources and their natural extensions to dependency and other deep annotation banks.

created a parallel treebank whose prototype includes ten typologically diverse languages and reflects a diverse set of phenomena. We thus present a unique, multilayered parallel treebank that represents more languages than are currently available in other treebanks, and different types of languages as well. It contains deep linguistic knowledge and allows for the parallel and simultaneous alignment of sentences at several levels. LFG’s f(unctional)-structure encodes dependency structures as well as information that is equivalent to Quasi-Logical Forms (van Genabith and Crouch, 1996). LFG’s c(onstituent)-structure provides information about constituency, hierarchical relations and part-of-speech. Currently, ParGramBank includes structures for the following languages (with the ISO 639-3 code and language family): English (*eng*, Indo-European), Georgian (*kat*, Kartvelian), German (*deu*, Indo-European), Hungarian (*hun*, Uralic), Indonesian (*ind*, Austronesian), Norwegian (Bokmål) (*nob*, Indo-European), Polish (*pol*, Indo-European), Turkish (*tur*, Altaic), Urdu (*urd*, Indo-European) and Wolof (*wol*, Niger-Congo). It is freely available for download under the CC-BY 3.0 license via the INESS treebanking environment and comes in two formats: a Prolog format and an XML format.<sup>2</sup>

This paper is structured as follows. Section 2 discusses related work in parallel treebanking. Section 3 presents ParGram and its approach to parallel treebanking. Section 4 focuses on the treebank design and its construction. Section 5 contains examples from the treebank, focusing on typological aspects and challenges for parallelism. Section 6 elaborates on the mechanisms for parallel alignment of the treebank.

## 2 Related Work

There have been several efforts in parallel treebanking across theories and annotation schemes.

Kuhn and Jellinghaus (2006) take a minimal approach towards multilingual parallel treebanking. They bootstrap phrasal alignments over a sentence-aligned parallel corpus of English, French, German and Spanish and report concrete treebank annotation work on a sample of sentences from the Europarl corpus. Their annotation

<sup>2</sup><http://iness.uib.no>. The treebank is in the public domain (CC-BY 3.0). The use of the INESS platform itself is not subject to any licensing. To access the treebank, click on ‘Treebank selection’ and choose the ParGram collection.

scheme is the “leanest” possible scheme in that it consists solely of a bracketing for a sentence in a language (where only those units that play the role of a semantic argument or modifier in a larger unit are bracketed) and a correspondence relation of the constituents across languages.

Klyueva and Mareček (2010) present a small parallel treebank using data and tools from two existing treebanks. They take a syntactically annotated gold standard text for one language and run an automated annotation on the parallel text for the other language. Manually annotated Russian data are taken from the SynTagRus treebank (Nivre et al., 2008), while tools for parsing the corresponding text in Czech are taken from the TectoMT framework (Popel and Žabokrtský, 2010).

The SMULTRON project is concerned with constructing a parallel treebank of English, German and Swedish. The sentences have been POS-tagged and annotated with phrase structure trees. These trees have been aligned on the sentence, phrase and word level. Additionally, the German and Swedish monolingual treebanks contain lemma information. The treebank is distributed in TIGER-XML format (Volk et al., 2010).

Megyesi et al. (2010) discuss a parallel English-Swedish-Turkish treebank. The sentences in each language are annotated morphologically and syntactically with automatic tools, aligned on the sentence and the word level and partially hand-corrected.<sup>3</sup>

A further parallel treebanking effort is ParTUT, a parallel treebank (Sanguinetti and Bosco, 2011; Bosco et al., 2012) which provides dependency structures for Italian, English and French and which can be converted to a CCG (Combinatory Categorical Grammar) format.

Closest to our work is the ParDeepBank, which is engaged in the creation of a highly parallel treebank of English, Portuguese and Bulgarian. ParDeepBank is couched within the linguistic framework of HPSG (Head-Driven Phrase Structure Grammar) and uses parallel automatic HPSG grammars, employing the same tools and implementation strategies across languages (Flickinger et al., 2012). The parallel treebank is aligned on the sentence, phrase and word level.

In sum, parallel treebanks have so far focused exclusively on Indo-European languages

<sup>3</sup>The paper mentions Hindi as the fourth language, but this is not yet available: <http://stp.lingfil.uu.se/~bea/turkiska/home-en.html>.

(with Turkish providing the one exception) and generally do not extend beyond three or four languages. In contrast, our ParGramBank treebank currently includes ten typologically different languages from six different language families (Altaic, Austronesian, Indo-European, Kartvelian, Niger-Congo, Uralic).

A further point of comparison with ParDeepBank is that it relies on dynamic treebanks, which means that structures are subject to change during the further development of the resource grammars. In ParDeepBank, additional machinery is needed to ensure correct alignment on the phrase and word level (Flickinger et al., 2012, p. 105). ParGramBank contains finalized analyses, structures and features that were designed collaboratively over more than a decade, thus guaranteeing a high degree of stable parallelism. However, with the methodology developed within XPAR, alignments can easily be recomputed from f-structure alignments in case of grammar or feature changes, so that we also have the flexible capability of allowing ParGramBank to include dynamic treebanks.

### 3 ParGram and its Feature Space

The ParGram grammars use the LFG formalism which produces c(onstituent)-structures (trees) and f(unctional)-structures as the syntactic analysis. LFG assumes a version of Chomsky’s Universal Grammar hypothesis, namely that all languages are structured by similar underlying principles (Chomsky, 1988; Chomsky, 1995). Within LFG, f-structures encode a language universal level of syntactic analysis, allowing for crosslinguistic parallelism at this level of abstraction. In contrast, c-structures encode language particular differences in linear word order, surface morphological vs. syntactic structures, and constituency (Dalrymple, 2001). Thus, while the Chomskyan framework is derivational in nature, LFG departs from this view by embracing a strictly representational approach to syntax.

ParGram tests the LFG formalism for its universality and coverage limitations to see how far parallelism can be maintained across languages. Where possible, analyses produced by the grammars for similar constructions in each language are parallel, with the computational advantage that the grammars can be used in similar applications and that machine translation can be simplified.

The ParGram project regulates the features and values used in its grammars. Since its inception in 1996, ParGram has included a “feature committee”, which collaboratively determines norms for the use and definition of a common multilingual feature and analysis space. Adherence to feature committee decisions is supported technically by a routine that checks the grammars for compatibility with a feature declaration (King et al., 2005); the feature space for each grammar is included in ParGramBank. ParGram also conducts regular meetings to discuss constructions, analyses and features.

For example, Figure 1 shows the c-structure of the Urdu sentence in (1) and the c-structure of its English translation. Figure 2 shows the f-structures for the same sentences. The left/upper c- and f-structures show the parse from the English ParGram grammar, the right/lower ones from Urdu ParGram grammar.<sup>4,5</sup> The c-structures encode linear word order and constituency and thus look very different; e.g., the English structure is rather hierarchical while the Urdu structure is flat (Urdu is a free word-order language with no evidence for a VP; Butt (1995)). The f-structures, in contrast, are parallel aside from grammar-specific characteristics such as the absence of grammatical gender marking in English and the absence of articles in Urdu.<sup>6</sup>

- (1) کسان نی اپنا ٹریکٹر بیچا ؟  
 kisAn=nE      apnA  
 farmer.M.Sg=Erg self.M.Sg  
 TrEkTar      bEc-A  
 tractor.M.Sg sell-Perf.M.Sg  
 ‘Did the farmer sell his tractor?’

With parallel analyses and parallel features, maximal parallelism across typologically different languages is maintained. As a result, during the construction of the treebank, post-processing and conversion efforts are kept to a minimum.

<sup>4</sup>The Urdu ParGram grammar makes use of a transliteration scheme that abstracts away from the Arabic-based script; the transliteration scheme is detailed in Malik et al. (2010).

<sup>5</sup>In the c-structures, dotted lines indicate distinct functional domains; e.g., in Figure 1, the NP *the farmer* and the VP *sell his tractor* belong to different f-structures: the former maps onto the SUBJ f-structure, while the latter maps onto the topmost f-structure (Dyvik et al., 2009). Section 6 elaborates on functional domains.

<sup>6</sup>The CASE feature also varies: since English does not distinguish between accusative, dative, and other oblique cases, the OBJ is marked with a more general *obl* CASE.

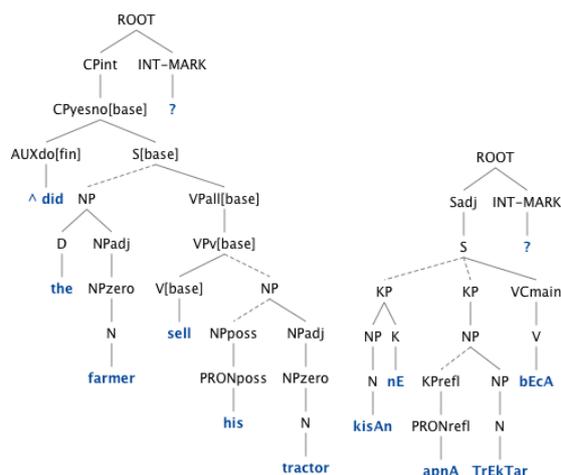


Figure 1: English and Urdu c-structures

We emphasize the fact that ParGramBank is characterized by a maximally reliable, human-controlled and linguistically deep parallelism across aligned sentences. Generally, the result of automatic sentence alignment procedures are parallel corpora where the corresponding sentences normally have the same purported meaning as intended by the translator, but they do not necessarily match in terms of structural expression. In building ParGramBank, conscious attention is paid to maintaining semantic and constructional parallelism as much as possible. This design feature renders our treebank reliable in cases when the constructional parallelism is reduced even at f-structure. For example, typological variation in the presence or absence of finite passive constructions represents a case of potential mismatch. Hungarian, one of the treebank languages, has no productive finite passives. The most common strategy in translation is to use an active construction with a topicalized object, with no overt subject and with 3PL verb agreement:

- (2) A fá-t      ki-vág-t-ák.  
 the tree-ACC out-cut-PAST-3PL  
 ‘The tree was cut down.’

In this case, a topicalized object in Hungarian has to be aligned with a (topical) subject in English. Given that both the sentence level and the phrase level alignments are human-controlled in the treebank (see sections 4 and 6), the greatest possible parallelism is reliably captured even in such cases of relative grammatical divergence.

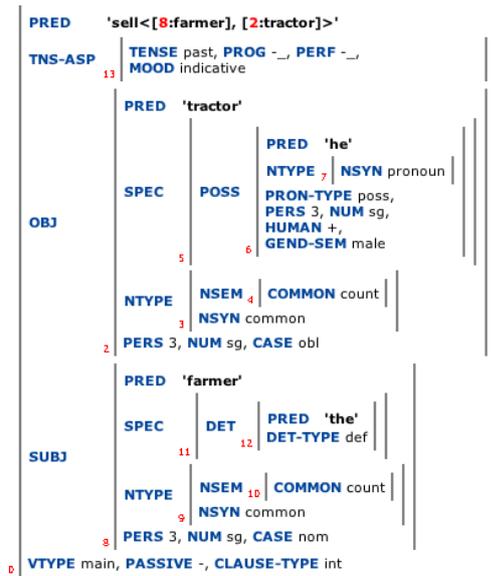


Figure 2: Parallel English and Urdu f-structures

#### 4 Treebank Design and Construction

For the initial seeding of the treebank, we focused on 50 sentences which were constructed manually to cover a diverse range of phenomena (transitivity, voice alternations, interrogatives, embedded clauses, copula constructions, control/raising verbs, etc.). We followed Lehmann et al. (1996) and Bender et al. (2011) in using coverage of grammatical constructions as a key component for grammar development. (3) lists the first 16 sentences of the treebank. An expansion to 100 sentences is scheduled for next year.

- (3) a. Declaratives:
1. The driver starts the tractor.
  2. The tractor is red.

- b. Interrogatives:
  3. What did the farmer see?
  4. Did the farmer sell his tractor?
- c. Imperatives:
  5. Push the button.
  6. Don't push the button.
- d. Transitivity:
  7. The farmer gave his neighbor an old tractor.
  8. The farmer cut the tree down.
  9. The farmer groaned.
- e. Passives and traditional voice:
  10. My neighbor was given an old tractor by the farmer.
  11. The tree was cut down yesterday.
  12. The tree had been cut down.
  13. The tractor starts with a shudder.
- f. Unaccusative:
  14. The tractor appeared.
- g. Subcategorized declaratives:
  15. The boy knows the tractor is red.
  16. The child thinks he started the tractor.

The sentences were translated from English into the other treebank languages. Currently, these languages are: English, Georgian, German, Hungarian, Indonesian, Norwegian (Bokmål), Polish, Turkish, Urdu and Wolof. The translations were done by ParGram grammar developers (i.e., expert linguists and native speakers).

The sentences were automatically parsed with ParGram grammars using XLE. Since the parsing was performed sentence by sentence, our resulting treebank is automatically aligned at the sentence level. The resulting c- and f-structures were banked in a database using the LFG Parsebanker (Rosén et al., 2009). The structures were disambiguated either prior to banking using XLE or during banking with the LFG Parsebanker and its discriminant-based disambiguation technique. The banked analyses can be exported and downloaded in a Prolog format using the LFG Parsebanker interface. Within XLE, we automatically convert the structures to a simple XML format and make these available via ParGramBank as well.

The Prolog format is used with applications which use XLE to manipulate the structures, e.g. for further semantic processing (Crouch and King, 2006) or for sentence condensation (Crouch et al., 2004).

## 5 Challenges for Parallelism

We detail some challenges in maintaining parallelism across typologically distinct languages.

### 5.1 Complex Predicates

Some languages in ParGramBank make extensive use of complex predicates. For example, Urdu uses a combination of predicates to express concepts that in languages like English are expressed with a single verb, e.g., ‘memory do’ = ‘remember’, ‘fear come’ = ‘fear’. In addition, verb+verb combinations are used to express permissive or aspectual relations. The strategy within ParGram is to abstract away from the particular surface morphosyntactic expression and aim at parallelism at the level of f-structure. That is, monoclausal predications are analyzed via a simple f-structure whether they consist of periphrastically formed complex predicates (Urdu, Figure 3), a simple verb (English, Figure 4), or a morphologically derived form (Turkish, Figure 5).

In Urdu and in Turkish, the top-level PRED is complex, indicating a composed predicate. In Urdu, this reflects the noun-verb complex predicate *sTArT kar* ‘start do’, in Turkish it reflects a morphological causative. Despite this morphosyntactic complexity, the overall dependency structure corresponds to that of the English simple verb.

(4) ڈرائیور ٹریکٹر کو سٹارٹ کرتا ہے

DrAIvar            TrEkTar=kO  
driver.M.Sg.Nom tractor.M.Sg=Acc  
sTArT    kartA        hE  
start.M.Sg do.Impf.M.Sg be.Pres.3Sg  
‘The driver starts the tractor.’

(5) sürücü traktör-ü çalış-tır-ıyor  
driver.Nom tractor-Acc work-Caus-Prog.3Sg  
‘The driver starts the tractor.’

The f-structure analysis of complex predicates is thus similar to that of languages which do not use complex predicates, resulting in a strong syntactic parallelism at this level, even across typologically diverse languages.

### 5.2 Negation

Negation also has varying morphosyntactic surface realizations. The languages in ParGramBank differ with respect to their negation strategies. Languages such as English and German use independent negation: they negate using words such as

<b>PRED</b>	'kar<[9:DrAIvar], 'sTArT<[4:TrEkTar]>'>'		
<b>VTYP</b>	13	COMPLEX-PRED-FORM kar, COMPLEX-PRED nv	
<b>TNS-ASP</b>	12	TENSE pres, MOOD indicative, ASPECT impf	
<b>LEX-SEM</b>	3	AGENTIVE +	
<b>OBJ</b>		<b>PRED</b> 'TrEkTar'	
		<b>SEM-PROP</b> 3 SPECIFIC +	
		<b>NTYPE</b>	<b>NSEM</b> 7 COMMON count
		6	<b>NSYN</b> common
	4	<b>PERS</b> 3, <b>NUM</b> sg, <b>GEN</b> masc, <b>CASE</b> acc	
<b>SUBJ</b>		<b>PRED</b> 'DrAIvar'	
		<b>NTYPE</b>	<b>NSEM</b> 11 COMMON count
		10	<b>NSYN</b> common
	9	<b>PERS</b> 3, <b>NUM</b> sg, <b>GEN</b> masc, <b>CASE</b> nom	
<b>CLAU</b>	8	CLAU-TYPE decl	

Figure 3: Complex predicate: Urdu analysis of (4)

<b>PRED</b>	'start<[7:driver], [2:tractor]>'		
<b>TNS-ASP</b>	12	TENSE pres, PROG -, PERF -, MOOD indicative	
<b>OBJ</b>		<b>PRED</b> 'tractor'	
		<b>SPEC</b>	<b>DET</b> 6 <b>PRED</b> 'the' DET-TYPE def
		<b>NTYPE</b>	<b>NSEM</b> 4 COMMON count
		3	<b>NSYN</b> common
	2	<b>PERS</b> 3, <b>NUM</b> sg, <b>CASE</b> obl	
<b>SUBJ</b>		<b>PRED</b> 'driver'	
		<b>SPEC</b>	<b>DET</b> 11 <b>PRED</b> 'the' DET-TYPE def
		<b>NTYPE</b>	<b>NSEM</b> 9 COMMON count
		8	<b>NSYN</b> common
	7	<b>PERS</b> 3, <b>NUM</b> sg, <b>CASE</b> nom	
<b>VTYP</b>	7	main, PASSIVE -, CLAU-TYPE decl	

Figure 4: Simple predicate: English analysis of (4)

adverbs (English *not*, German *nicht*) or verbs (English *do*-support). Other languages employ non-independent, morphological negation techniques; Turkish, for instance, uses an affix on the verb, as in (6).

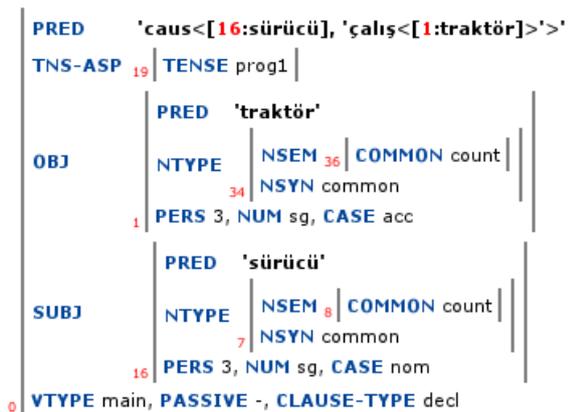


Figure 5: Causative: Turkish analysis of (5)

- (6) düğme-ye bas-ma  
 button-Dat push-Neg.Imp  
 'Don't push the button.'

Within ParGram we have not abstracted away from this surface difference. The English *not* in (6) functions as an adverbial adjunct that modifies the main verb (see top part of Figure 6) and information would be lost if this were not represented at f-structure. However, the same cannot be said of the negative affix in Turkish — the morphological affix is not an adverbial adjunct. We have therefore currently analyzed morphological negation as adding a feature to the f-structure which marks the clause as negative, see bottom half of Figure 6.

### 5.3 Copula Constructions

Another challenge to parallelism comes from copula constructions. An approach advocating a uniform treatment of copulas crosslinguistically was advocated in the early years of ParGram (Butt et al., 1999b), but this analysis could not do justice to the typological variation found with copulas. ParGramBank reflects the typological difference with three different analyses, with each language making a language-specific choice among the three possibilities that have been identified (Dalrymple et al., 2004; Nordlinger and Sadler, 2007; Attia, 2008; Sulger, 2011; Laczkó, 2012).

The possible analyses are demonstrated here with respect to the sentence *The tractor is red*. The English grammar (Figure 7) uses a raising approach that reflects the earliest treatments of copulas in LFG (Bresnan, 1982). The copula takes a non-finite complement whose subject is raised to the matrix clause as a non-thematic subject of the copula. In contrast, in Urdu (Figure 8), the

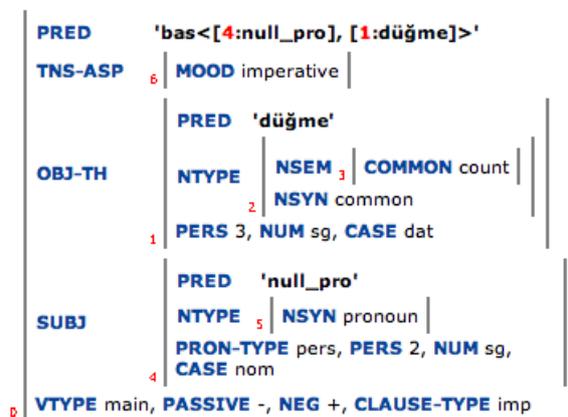
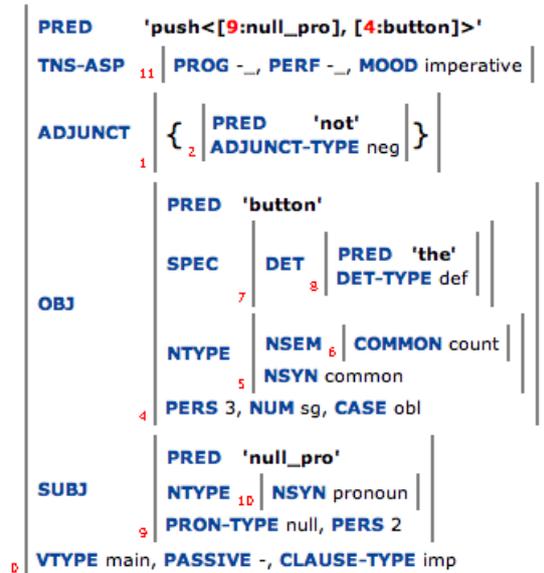


Figure 6: Different f-structural analyses for negation (English vs. Turkish)

copula is a two-place predicate, assigning SUBJ and PREDLINK functions. The PREDLINK function is interpreted as predicating something about the subject. Finally, in languages like Indonesian (Figure 9), there is no overt copula and the adjective is the main predicational element of the clause.

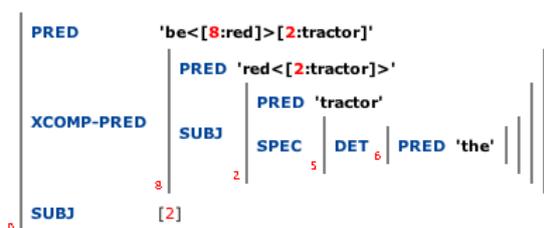


Figure 7: English copula example



Figure 8: Urdu copula example

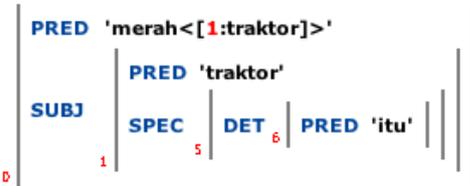


Figure 9: Indonesian copula example

#### 5.4 Summary

This section discussed some challenges for maintaining parallel analyses across typologically diverse languages. Another challenge we face is when no corresponding construction exists in a language, e.g. with impersonals as in the English *It is raining*. In this case, we provide a translation and an analysis of the structure of the corresponding translation, but note that the phenomenon being exemplified does not actually exist in the language. A further extension to the capabilities of the treebank could be the addition of pointers from the alternative structure used in the translation to the parallel aligned set of sentences that correspond to this alternative structure.

### 6 Linguistically Motivated Alignment

The treebank is automatically aligned on the sentence level, the top level of alignment within ParGramBank. For phrase-level alignments, we use the drag-and-drop alignment tool in the LFG Parsebanker (Dyvik et al., 2009). The tool allows the alignment of f-structures by dragging the index of a subsidiary source f-structure onto the index of the corresponding target f-structure. Two f-structures correspond if they have translationally matching predicates, and the arguments of each predicate correspond to an argument or adjunct in the other f-structure. The tool automatically computes the alignment of c-structure nodes on the basis of the manually aligned corresponding f-structures.<sup>7</sup>

<sup>7</sup>Currently we have not measured inter-annotator agreement (IAA) for the f-structure alignments. The f-structure alignments were done by only one person per language pair. We anticipate that multiple annotators will be needed for this

This method is possible because the c-structure to f-structure correspondence (the  $\phi$  relation) is encoded in the ParGramBank structures, allowing the LFG Parsebanker tool to compute which c-structure nodes contributed to a given f-structure via the inverse ( $\phi^{-1}$ ) mapping. A set of nodes mapping to the same f-structure is called a ‘functional domain’. Within a source and a target functional domain, two nodes are automatically aligned only if they dominate corresponding word forms. In Figure 10 the nodes in each functional domain in the trees are connected by whole lines while dotted lines connect different functional domains. Within a functional domain, thick whole lines connect the nodes that share alignment; for simplicity the alignment is only indicated for the top nodes. The automatically computed c-structural alignments are shown by the curved lines. The alignment information is stored as an additional layer and can be used to explore alignments at the string (word), phrase (c-)structure, and functional (f-)structure levels.

We have so far aligned the treebank pairs English-Urdu, English-German, English-Polish and Norwegian-Georgian. As Figure 10 illustrates for (7) in an English-Urdu pairing, the English object *neighbor* is aligned with the Urdu indirect object (OBJ-GO) *hamsAyA* ‘neighbor’, while the English indirect object (OBJ-TH) *tractor* is aligned with the Urdu object *TrEkTar* ‘tractor’. The c-structure correspondences were computed automatically from the f-structure alignments.

- (7) کسان ني اپني همسايي کو پرانا ٹريڪٽر ديا  
 kisAn=nE                    apnE  
 farmer.M.Sg=Erg self.Obl  
 hamsAyE=kO                    purAnA  
 neighbor.M.Sg.Obl=Acc old.M.Sg  
 TrEkTar                    di-yA  
 tractor.M.Sg give-Perf.M.Sg  
 ‘The farmer gave his neighbor an old tractor.’

The INESS platform additionally allows for the highlighting of connected nodes via a mouse-over technique. It thus provides a powerful and flexible tool for the semi-automatic alignment and subse-

task in the future, in which case we will measure IAA for this step.

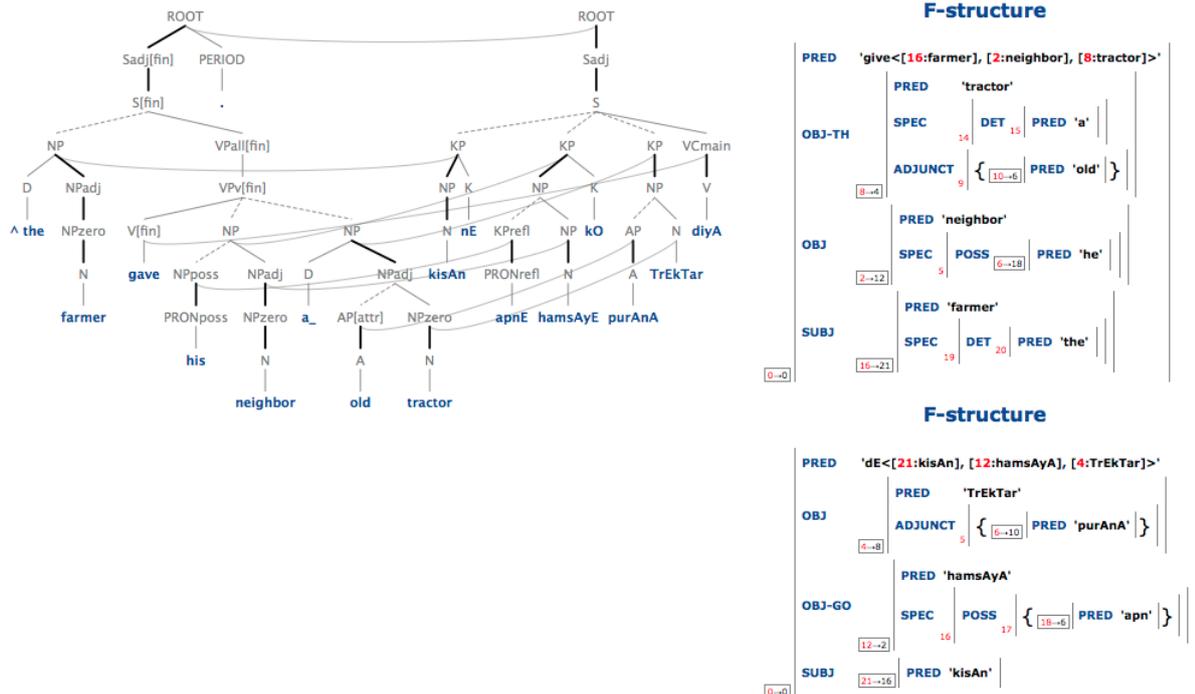


Figure 10: Phrase-aligned treebank example English-Urdu: *The farmer gave his neighbor an old tractor.*

quent inspection of parallel treebanks which contain highly complex linguistic structures.<sup>8</sup>

## 7 Discussion and Future Work

We have discussed the construction of ParGramBank, a parallel treebank for ten typologically different languages. The analyses in ParGramBank are the output of computational LFG ParGram grammars. As a result of ParGram’s centrally agreed upon feature sets and prototypical analyses, the representations are not only deep in nature, but maximally parallel. The representations offer information about dependency relations as well as word order, constituency and part-of-speech.

In future ParGramBank releases, we will provide more theory-neutral dependencies along with the LFG representations. This will take the form of triples (King et al., 2003). We also plan to provide a POS-tagged and a named entity marked up version of the sentences; these will be of use for more general NLP applications and for systems which use such markup as input to deeper processing.

<sup>8</sup>One reviewer inquires about possibilities of linking (semi-)automatically between languages, for example using lexical resources such as WordNets or Panlex. We agree that this would be desirable, but unrealizable, since many of the languages included in ParGramBank do not have a WordNet resource and are not likely to achieve an adequate one soon.

Third, the treebank will be expanded to include 100 more sentences within the next year. We also plan to include more languages as other ParGram groups contribute structures to ParGramBank.

ParGramBank, including its multilingual sentences and all annotations, is made freely available for research and commercial use under the CC-BY 3.0 license via the INESS platform, which supports alignment methodology developed in the XPAR project and provides search and visualization methods for parallel treebanks. We encourage the computational linguistics community to contribute further layers of annotation, including semantic (Crouch and King, 2006), abstract knowledge representational (Bobrow et al., 2007), PropBank (Palmer et al., 2005), or TimeBank (Mani and Pustejovsky, 2004) annotations.

## References

- Mohammed Attia. 2008. A Unified Analysis of Copula Constructions. In *Proceedings of the LFG ’08 Conference*, pages 89–108. CSLI Publications.
- Emily M. Bender, Dan Flickinger, and Stephan Oepen. 2011. Grammar Engineering and Linguistic Hypothesis Testing: Computational Support for Complexity in Syntactic Analysis. In Emily M. Bender and Jennifer E. Arnold, editors, *Languages from a Cognitive Perspective: Grammar, Usage and Processing*, pages 5–30. CSLI Publications.

- Daniel G. Bobrow, Cleo Condoravdi, Dick Crouch, Valeria de Paiva, Lauri Karttunen, Tracy Holloway King, Rowan Nairn, Lottie Price, and Annie Zaenen. 2007. Precision-focused Textual Inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.
- Cristina Bosco, Manuela Sanguinetti, and Leonardo Lesmo. 2012. The Parallel-TUT: a multilingual and multiformat treebank. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 1932–1938, Istanbul, Turkey. European Language Resources Association (ELRA).
- Joan Bresnan. 1982. The Passive in Lexical Theory. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*, pages 3–86. The MIT Press.
- Joan Bresnan. 2001. *Lexical-Functional Syntax*. Blackwell Publishing.
- Miriam Butt, Stefanie Dipper, Anette Frank, and Tracy Holloway King. 1999a. Writing Large-Scale Parallel Grammars for English, French and German. In *Proceedings of the LFG99 Conference*. CSLI Publications.
- Miriam Butt, Tracy Holloway King, María-Eugenia Niño, and Frédérique Segond. 1999b. *A Grammar Writer's Cookbook*. CSLI Publications.
- Miriam Butt, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi, and Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of the COLING-2002 Workshop on Grammar Engineering and Evaluation*, pages 1–7.
- Miriam Butt. 1995. *The Structure of Complex Predicates in Urdu*. CSLI Publications.
- Noam Chomsky. 1988. *Lectures on Government and Binding: The Pisa Lectures*. Foris Publications.
- Noam Chomsky. 1995. *The Minimalist Program*. MIT Press.
- Dick Crouch and Tracy Holloway King. 2006. Semantics via F-structure Rewriting. In *Proceedings of the LFG06 Conference*, pages 145–165. CSLI Publications.
- Dick Crouch, Tracy Holloway King, John T. Maxwell III, Stefan Riezler, and Annie Zaenen. 2004. Exploiting F-structure Input for Sentence Condensation. In *Proceedings of the LFG04 Conference*, pages 167–187. CSLI Publications.
- Dick Crouch, Mary Dalrymple, Ronald M. Kaplan, Tracy Holloway King, John T. Maxwell III, and Paula Newman, 2012. *XLE Documentation*. Palo Alto Research Center.
- Mary Dalrymple, Helge Dyvik, and Tracy Holloway King. 2004. Copular Complements: Closed or Open? In *Proceedings of the LFG '04 Conference*, pages 188–198. CSLI Publications.
- Mary Dalrymple. 2001. *Lexical Functional Grammar*, volume 34 of *Syntax and Semantics*. Academic Press.
- Helge Dyvik, Paul Meurer, Victoria Rosén, and Koenraad De Smedt. 2009. Linguistically Motivated Parallel Parsebanks. In *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories (TLT8)*, pages 71–82, Milan, Italy. EDU-Catt.
- Dan Flickinger, Valia Kordoni, Yi Zhang, António Branco, Kiril Simov, Petya Osenova, Catarina Carvalho, Francisco Costa, and Sérgio Castro. 2012. ParDeepBank: Multiple Parallel Deep Treebanking. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories (TLT11)*, pages 97–107, Lisbon. Edições Colibri.
- Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald Kaplan. 2003. The PARC700 Dependency Bank. In *Proceedings of the EACL03: 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*.
- Tracy Holloway King, Martin Forst, Jonas Kuhn, and Miriam Butt. 2005. The Feature Space in Parallel Grammar Writing. In Emily M. Bender, Dan Flickinger, Frederik Fouvry, and Melanie Siegel, editors, *Research on Language and Computation: Special Issue on Shared Representation in Multilingual Grammar Engineering*, volume 3, pages 139–163. Springer.
- Natalia Klyueva and David Mareček. 2010. Towards a Parallel Czech-Russian Dependency Treebank. In *Proceedings of the Workshop on Annotation and Exploitation of Parallel Corpora*, Tartu. Northern European Association for Language Technology (NEALT).
- Jonas Kuhn and Michael Jellinghaus. 2006. Multilingual Parallel Treebanking: A Lean and Flexible Approach. In *Proceedings of the LREC 2006*, Genoa, Italy. ELRA/ELDA.
- Tibor Laczkó. 2012. On the (Un)Bearable Lightness of Being an LFG Style Copula in Hungarian. In *Proceedings of the LFG12 Conference*, pages 341–361. CSLI Publications.
- Sabine Lehmann, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Hervé Compagnon, Judith Baur, Lorna Balkan, and Doug Arnold. 1996. TSNLP — Test Suites for Natural Language Processing. In *Proceedings of COLING*, pages 711–716.
- Muhammad Kamran Malik, Tafseer Ahmed, Sebastian Sulger, Tina Bögel, Atif Gulzar, Ghulam Raza, Sarmad Hussain, and Miriam Butt. 2010. Transliterating Urdu for a Broad-Coverage Urdu/Hindi LFG Grammar. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC 2010)*, Valletta, Malta.

- Inderjeet Mani and James Pustejovsky. 2004. Temporal Discourse Models for Narrative Structure. In *Proceedings of the 2004 ACL Workshop on Discourse Annotation*, pages 57–64.
- Beáta Megyesi, Bengt Dahlqvist, Éva Á. Csató, and Joakim Nivre. 2010. The English-Swedish-Turkish Parallel Treebank. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).
- Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective Sharing for Multilingual Dependency Parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 629–637, Jeju Island, Korea, July. Association for Computational Linguistics.
- Joakim Nivre, Igor Boguslavsky, and Leonid Iomdin. 2008. Parsing the SynTagRus Treebank. In *Proceedings of COLING08*, pages 641–648.
- Rachel Nordlinger and Louisa Sadler. 2007. Verbless Clauses: Revealing the Structure within. In Annie Zaenen, Jane Simpson, Tracy Holloway King, Jane Grimshaw, Joan Maling, and Chris Manning, editors, *Architectures, Rules and Preferences: A Festschrift for Joan Bresnan*, pages 139–160. CSLI Publications.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.
- Martin Popel and Zdeněk Žabokrtský. 2010. TectoMT: Modular NLP Framework. In *Proceedings of the 7th International Conference on Advances in Natural Language Processing (IceTAL 2010)*, pages 293–304.
- Victoria Rosén, Paul Meurer, and Koenraad de Smedt. 2009. LFG Parsebanker: A Toolkit for Building and Searching a Treebank as a Parsed Corpus. In *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories (TLT7)*, pages 127–133, Utrecht. LOT.
- Victoria Rosén, Koenraad De Smedt, Paul Meurer, and Helge Dyvik. 2012. An Open Infrastructure for Advanced Treebanking. In *META-RESEARCH Workshop on Advanced Treebanking at LREC2012*, pages 22–29, Istanbul, Turkey.
- Manuela Sanguinetti and Cristina Bosco. 2011. Building the Multilingual TUT Parallel Treebank. In *Proceedings of Recent Advances in Natural Language Processing*, pages 19–28.
- Sebastian Sulger. 2011. A Parallel Analysis of have-Type Copular Constructions in have-Less Indo-European Languages. In *Proceedings of the LFG '11 Conference*. CSLI Publications.
- Josef van Genabith and Dick Crouch. 1996. Direct and Underspecified Interpretations of LFG f-structures. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, volume 1, pages 262–267, Copenhagen, Denmark.
- Martin Volk, Anne Göhring, Torsten Marek, and Yvonne Samuelsson. 2010. SMULTRON (version 3.0) — The Stockholm MULTilingual parallel TReebank. [http://www.cl.uzh.ch/research/paralleltreebanks\\_en.html](http://www.cl.uzh.ch/research/paralleltreebanks_en.html).

# Identifying Bad Semantic Neighbors for Improving Distributional Thesauri

Olivier Ferret

CEA, LIST, Vision and Content Engineering Laboratory,  
Gif-sur-Yvette, F-91191 France.  
olivier.ferret@cea.fr

## Abstract

Distributional thesauri are now widely used in a large number of Natural Language Processing tasks. However, they are far from containing only interesting semantic relations. As a consequence, improving such thesaurus is an important issue that is mainly tackled indirectly through the improvement of semantic similarity measures. In this article, we propose a more direct approach focusing on the identification of the neighbors of a thesaurus entry that are not semantically linked to this entry. This identification relies on a discriminative classifier trained from unsupervised selected examples for building a distributional model of the entry in texts. Its bad neighbors are found by applying this classifier to a representative set of occurrences of each of these neighbors. We evaluate the interest of this method for a large set of English nouns with various frequencies.

## 1 Introduction

The work we present in this article focuses on the automatic building of a thesaurus from a corpus. As illustrated by Table 1, such thesaurus gives for each of its entries a list of words, called *semantic neighbors*, that are supposed to be semantically linked to the entry. Generally, each neighbor is associated with a weight that characterizes the strength of its link with the entry and all the neighbors of an entry are sorted according to the decreasing order of their weight.

The term *semantic neighbor* is very generic and can have two main interpretations according to the kind of semantic relations it is based on: one relies only on paradigmatic relations, such as hypernymy or synonymy, while the other consid-

ers syntagmatic relations, called collocation relations by (Halliday and Hasan, 1976) in the context of lexical cohesion or “non-classical relations” by (Morris and Hirst, 2004). The distinction between these two interpretations refers to the distinction between the notions of *semantic similarity* and *semantic relatedness* as it was done in (Budanitsky and Hirst, 2006) or in (Zesch and Gurevych, 2010) for instance. However, the limit between these two notions is sometimes hard to find in existing work as terms *semantic similarity* and *semantic relatedness* are often used interchangeably. Moreover, *semantic similarity* is frequently considered as included into *semantic relatedness* and the two problems are often tackled by using the same methods. In the remainder of this article, we will use the term *semantic similarity* with its generic sense and the term *semantic relatedness* for referring more specifically to similarity based on syntagmatic relations.

Following work such as (Grefenstette, 1994), a widespread way to build a thesaurus from a corpus is to use a semantic similarity measure for extracting the semantic neighbors of the entries of the thesaurus. Three main ways of implementing such measures can be distinguished. The first one relies on handcrafted resources in which semantic relations are clearly identified. Work based on WordNet-like lexical networks for building semantic similarity measures such as (Budanitsky and Hirst, 2006) or (Pedersen et al., 2004) falls into this category. These measures typically exploit the hierarchical structure of these networks, based on hypernymy relations. The second approach makes use of a less structured source of knowledge about words such as the definitions of classical dictionaries or the *glosses* of WordNet. WordNet’s *glosses* were used to support Lesk-like measures in (Banerjee and Pedersen, 2003) and more recently, measures were also defined from Wikipedia or Wiktionaries (Gabrilovich and

Markovitch, 2007). The last option is the corpus-based approach, based on the distributional hypothesis (Firth, 1957): each word is characterized by the set of contexts from a corpus in which it appears and the semantic similarity of two words is computed from the contexts they share. This perspective was first adopted by (Grefenstette, 1994) and (Lin, 1998) and then, explored in details in (Curran and Moens, 2002b), (Weeds, 2003) or (Heylen et al., 2008).

The problem of improving the results of the “classical” implementation of the distributional approach as it can be found in (Curran and Moens, 2002a) for instance was already tackled by some work. A part of these proposals focus on the weighting of the elements that are part of the contexts of words such as (Broda et al., 2009), in which the weights of context elements are turned into ranks, or (Zhitomirsky-Geffet and Dagan, 2009), followed and extended by (Yamamoto and Asakura, 2010), that proposes a bootstrapping method for modifying the weights of context elements according to the semantic neighbors found by an initial distributional similarity measure. However, another part of these proposals implies more radical changes. The use of dimensionality reduction techniques, for instance Latent Semantic Analysis in (Padó and Lapata, 2007), the multi-prototype (Reisinger and Mooney, 2010) or exemplar-based models (Erk and Pado, 2010), the Deep Learning approach of (Huang et al., 2012) or the redefinition of the distributional approach in a Bayesian framework (Kazama et al., 2010) can be classified into this second category.

The work we present in this article takes place in the framework defined by (Grefenstette, 1994) for implementing the distributional approach but proposes a new method for improving a thesaurus built in this context based on the identification of its bad semantic neighbors rather than on the adaptation of the weight of their features.

## 2 Principles

Our work shares with (Zhitomirsky-Geffet and Dagan, 2009) the use of a kind of bootstrapping as it starts from a distributional thesaurus and to some extent, exploits it for its improvement. However, it adopts a more indirect approach: instead of selecting the “best” semantic neighbors of an entry in the thesaurus for adapting the weights of distributional context elements, it focuses on the detection

of its bad semantic neighbors, that is to say the neighbors of the entry that are actually not semantically similar to the entry. In Table 1, *waterworks* for the entry *cabdriver* and *hollowness* for the entry *machination* are two examples of such kind of neighbors. By discarding these bad neighbors or at least by downgrading them, the rank of true semantic neighbors is expected to be lower. This makes the thesaurus more interesting to use since the quality of such thesaurus strongly decreases as the rank of the neighbors of its entries increases (see Section 4.1 for an illustration), which means in practice that only the first neighbors of an entry can be generally exploited.

The approach we propose for identifying the bad semantic neighbors of a thesaurus entry relies on the distributional hypothesis, as the method for the initial building of the thesaurus, but implements it in a different way. This hypothesis roughly specifies that from a semantic viewpoint, the meaning of a word can be characterized by the set of contexts in which this word occurs. As a consequence, two words are considered as semantically similar if they occur in a large enough set of shared contexts. In work such as (Curran and Moens, 2002a), this hypothesis is implemented by collecting for each entry the words it co-occurs with in a large corpus. This co-occurrence can be based either on the position of the word in the text in relation to the entry or on the presence of a syntactic relation between the entry and the word. As a result, the distributional representation of a word takes the unstructured form of a bag of words or the more structured form of a set of pairs {syntactic relation, word}. A variant of this approach was proposed in (Kazama et al., 2010) where the distributional representation of a word is modeled as a multinomial distribution with Dirichlet as prior.

However, this approach globally faces a certain lack of diversity and complexity of the features of its models. For instance, features such as ngrams of words or ngrams of parts of speech are not considered whereas they are widely used in tasks such as word sense disambiguation (WSD) for instance, probably because they would lead to very large models and because similarity measures such as the *Cosine* measure are not necessarily suitable for heterogeneous representations (Alexandrescu and Kirchhoff, 2007). Hence, we propose in this article to build a discriminative model for repre-

abnormality	defect [0.30], disorder [0.23], deformity [0.22], mutation [0.21], prolapse [0.21], anomaly [0.21] ...
agreement	accord [0.44], deal [0.41], pact [0.38], treaty [0.36], negotiation [0.35], proposal [0.32], arrangement [0.30] ...
cabdriver	waterworks [0.23], toolmaker [0.22], weaponeer [0.17], valkyry [0.17], wang [0.17], amusement-park [0.17] ...
machination	hollowness [0.15], share-price [0.12], clockmaker [0.12], huguenot [0.12], wrangling [0.12], alternation [0.12] ...

Table 1: First neighbors of some entries of the distributional thesaurus of section 3.2

senting the contexts of a word since this kind of models are known to integrate easily a wide set of different types of features. This model aims more precisely at discriminating from a semantic viewpoint a word in context, *i.e.* in a sentence, from all other words and more particularly, from those of its neighbors in a distributional thesaurus that are likely to be actually not semantically similar to it. The underlying hypothesis follows the distributional principles: a word and a synonym should appear in the same contexts, which means that they are characterized by the same features. As a consequence, a model based on these features that can identify a word in a sentence is likely to identify also a synonym of this word in a sentence, and by extension, to identify a word that is paradigmatically linked to it. More precisely, we found that such model is specifically effective for discarding the bad neighbors of the entries of a distributional thesaurus.

### 3 Improving a distributional thesaurus

#### 3.1 Overview

The principles presented in the previous section face one major problem compared to the “classical” distributional approach : the semantic similarity of two words can be evaluated directly by computing the similarity of their distributional representations. However, in our case, since this representation is a discriminative model, the similarity of two words can not be evaluated through the direct comparison of their models. These models have to be applied to words in context for being exploited. As a consequence, for deciding whether a neighbor of a thesaurus entry is a bad neighbor or not, the discriminative model of the entry has to be applied to occurrences of this neighbor in texts. Hence, the method we propose for improving a distributional thesaurus applies the following process to each of its entries:

- building of a classifier for determining whether a word in a sentence corresponds or not to the entry;
- selection of a set of examples sentences for each of the neighbors of the entry in the the-

saurus;

- application of the classifier to these sentences;
- identification of bad neighbors according to the results of the classifier;
- reranking of entry’s neighbors according to bad neighbors.

#### 3.2 Building of the initial thesaurus

Before introducing our method for improving distributional thesauri, we first present the way we build such a thesaurus. As in (Lin, 1998) or (Curran and Moens, 2002a), this building is based on the definition of a semantic similarity measure from a corpus. The corpus used for defining this measure was the AQUAINT-2 corpus, a middle-size corpus made of around 380 million words coming from news articles. Although our target language is English, we chose to limit deliberately the level of the tools applied for preprocessing texts to part-of-speech tagging and lemmatization to make possible the transposition of our method to a large set of languages. This seems to be a reasonable compromise between the approach of (Freitag et al., 2005), in which none normalization of words is done, and the more widespread use of syntactic parsers in work such as (Lin, 1998). More precisely, we used *TreeTagger* (Schmid, 1994) for performing the linguistic preprocessing of the AQUAINT-2 corpus.

For the extraction of distributional data and the characteristics of the distributional similarity measure, we adopted the options of (Ferret, 2010), resulting from a kind of grid search procedure performed with the extended TOEFL test proposed in (Freitag et al., 2005) as an optimization objective. More precisely, the following characteristics were taken:

- distributional contexts made of the co-occurrences collected in a 3 word window centered on each occurrence in the corpus of the target word. These co-occurrences were restricted to nouns, verbs and adjectives;
- soft filtering of contexts: removal of co-occurrences with only one occurrence;
- weighting function of co-occurrences in con-

texts = *Pointwise Mutual Information* (PMI) between the target word and the co-occurent;

- similarity measure between contexts, for evaluating the semantic similarity of two words = *Cosine* measure.

The building of our initial thesaurus from the similarity measure above was performed classically by extracting the closest semantic neighbors of each of its entries. More precisely, the selected measure was computed between each entry and its possible neighbors. These neighbors were then ranked in the decreasing order of the values of this measure and the first 100 neighbors were kept as the semantic neighbors of the entry. Both entries and possible neighbors were AQUAINT-2 nouns whose frequency was higher than 10.

### 3.3 Building a discriminative model of words in context

As mentioned in section 3.1, the starting point of our reranking process is the definition of a model for determining to what extent a word in a sentence, which is not supposed to be known in the context of this task, corresponds or not to a reference word  $E$ . This task can also be viewed as a tagging task in which the occurrences of a target word  $T$  are labeled with two tags:  $E$  and  $notE$ . In the context of our global objective, we are not of course interested by this task itself but rather by the fact that such classifier is likely to model the contexts in which  $E$  occurs and as a consequence, is also likely to model its meaning according to the distributional hypothesis.

A step further, such classifier can be viewed as a means for testing whether or not a word has the same meaning as  $E$ . This is a problem close to WSD as it is performed in the context of the pseudo-word disambiguation paradigm (Gale et al., 1992): a pseudo-word is created with two senses,  $E$  and  $notE$ ,  $notE$  corresponding to one or several words that are supposed to be representative of a meaning different from the meaning of  $E$ . The objective is then to build a classifier for distinguishing the pseudo-senses  $E$  and  $notE$ . As a consequence of this view, we adopt the same kind of features as the ones used for WSD for building our classifier. More precisely, we follow (Lee and Ng, 2002), a reference work for WSD, by adopting a Support Vector Machines (SVM) classifier with a linear kernel and three kinds of features for characterizing each considered occur-

rence in a text of the reference word  $E$ :

- neighboring words;
- Part-of-Speech (POS) of neighboring words;
- local collocations.

Only features based on syntactic relations are not taken from (Lee and Ng, 2002) since their use would have not been coherent with the window based approach of the building of our initial thesaurus.

For the *neighboring words* features, we consider all plain words (common and proper nouns, verbs and adjectives) and adverbs that are present in the same sentence of an occurrence of  $E$ . Each neighboring word is represented under its lemma form as a binary feature whose value is equal to 1 when it is present in the same sentence as  $E$ .

For the second type of features, we take more precisely the POS of the three words before  $E$  and those of the three words after  $E$ . Each pair {POS, position} corresponds to a binary feature for the SVM classifier. A special *empty* symbol is used for the POS when the position goes beyond the end or the beginning of the current sentence. Since we analyze texts with *TreeTagger*, the tagset is very close to the set of Penn Treebank tags.

Finally, the *local collocations* features correspond to pairs of words, named collocations, in the neighborhood of  $E$ . A collocation is specified by the notation  $C_{i,j}$ , with  $i$  and  $j$  referring to the position of the first and the second word of the collocation. In our case,  $i$  and  $j$  take their values in the interval  $[-3, +3]$ , similarly to POS. More precisely, the following 11 types of collocations are extracted for each occurrence of  $E$ :  $C_{-1,-1}$ ,  $C_{1,1}$ ,  $C_{-2,-2}$ ,  $C_{2,2}$ ,  $C_{-2,-1}$ ,  $C_{-1,1}$ ,  $C_{1,2}$ ,  $C_{-3,-1}$ ,  $C_{-2,1}$ ,  $C_{-1,2}$  and  $C_{1,3}$ . As for POS, a special empty symbol stands for words beyond the end or the beginning of the sentence and similarly to *neighboring words* features, words in collocations are given under their lemma form. Each instance of the 11 types of collocations is represented by a tuple (lemma1, position1, lemma2, position2) and leads to a binary feature for the SVM classifier.

In accordance with the process of section 3.1, a specific SVM classifier is trained for each entry of our initial thesaurus, which requires the unsupervised selection of a set of positive and negative examples. The case of positive examples is simple: a fixed number of sentences containing at least one occurrence of the target entry are randomly chosen in the corpus used for building our

initial thesaurus and the first occurrence of this entry in the sentence is taken as a positive example. Since we want to characterize words as much as possible from a semantic viewpoint, the selection of negative examples is guided by our initial thesaurus. Choosing a neighbor of the entry with a high rank would guarantee in principle few false negative examples, that is to say words<sup>1</sup> which are semantically similar to the entry, since the number of such neighbors strongly decreases as the rank of neighbors increases as we will illustrate it in section 4.1. In practice, taking neighbors with a rather small rank as negative examples is a better option because these examples are more useful in terms of discrimination as they are close to the transition zone between negative and positive examples. Moreover, in order to limit the risk of selecting only false negative examples, three neighbors are taken as negative examples, at ranks 10, 15 and 20<sup>2</sup>. For each of these negative examples, a fixed number of sentences is selected following the same principles as for positive examples, which means that on average, the number of negative examples is equal to three times the number of positive examples. This ratio reflects the fact that among the neighbors of an entry, the number of those that are semantically similar to the entry is far lower than the number of those that are not.

### 3.4 Identification of bad neighbors and thesaurus reranking

Once a word-in-context classifier was trained for an entry, it is used for identifying the bad neighbors of this entry, that is to say the neighbors that are not semantically similar to it. As this classifier can only be applied to words in context, a fixed number of representative occurrences have to be selected from our reference corpus for each neighbor of the entry. This selection is performed similarly to the selection of positive and negative examples in the previous section. The application of our word-in-context classifier to each of these occurrences determines whether the context of this occurrence is likely to be compatible with the context of an occurrence of the entry.

In practice, the decision of the classifier is rarely

<sup>1</sup>More precisely, an example here is an occurrence of a word in a text but by extension, we also use the term *example* for referring to the word itself.

<sup>2</sup>It should be noted that these ranks come from the evaluation of section 4.1 but their choice is not the result of an optimization process.

positive, which is not surprising: even if two words are semantically equivalent, each one is characterized by specific usages, especially in a given corpus, and some features of our classifier, such as the collocation features, are more likely to capture such specificities than the unigrams of “classical” distributional contexts. As a consequence, we consider that a positive outcome of our classifier is a significant hint about the presence of a word that is semantically similar to the entry and we keep a neighbor as a “good” neighbor if at least a fixed number  $G$  of its occurrences, among those selected as reference, are tagged positively by our word-in-context classifier. Conversely, a neighbor is defined as “bad” if the number of its reference occurrences tagged positively by our classifier is lower or equal to  $G$ .

The neighbors of an entry identified as bad neighbors are not fully discarded. They are rather downgraded to the end of the list of neighbors. Among the downgraded neighbors, their initial order is left unchanged. It should be noted that the word-in-context classifier is not applied to the neighbors whose occurrences are used for its training as it would frequently lead to downgrade these neighbors, which is not necessarily optimum as we chose them with a rather low rank.

## 4 Experiments and evaluation

### 4.1 Initial thesaurus evaluation

Table 2 shows the results of the evaluation of our initial thesaurus, achieved by comparing the selected semantic neighbors with two complementary reference resources: WordNet 3.0 synonyms (Miller, 1990) [W], which characterize a semantic similarity based on paradigmatic relations, and the Moby thesaurus (Ward, 1996) [M], which gathers a larger set of types of relations and is more representative of *semantic relatedness*<sup>3</sup>. The fourth column of Table 2, which gives the average number of synonyms and similar words in our references for the AQUAINT-2 nouns, also illustrates the difference of these two resources in terms of richness. A fusion of the two resources is also considered [WM]. As our objective is to evaluate the extracted semantic neighbors and not the ability to rebuild the reference resources, these re-

<sup>3</sup>The Moby thesaurus includes more precisely both paradigmatic and syntactic relations but we will sometimes use the term *synonym* as a shortcut for referring to all the words associated to one of its entries.

freq.	ref.	#eval. words	#syn. / word	recall	R-prec.	MAP	P@1	P@5	P@10	P@100
all # 14,670	W	10,473	2.9	24.6	8.2	9.8	11.7	5.1	3.4	0.7
	M	9,216	50.0	9.5	6.7	3.2	24.1	16.4	13.0	4.8
	WM	12,243	38.7	9.8	7.7	5.6	22.5	14.1	10.8	3.8
high # 4,378	W	3,690	3.7	28.3	11.1	12.5	17.2	7.7	5.1	1.0
	M	3,732	69.4	11.4	10.2	4.9	41.3	28.0	21.9	7.9
	WM	4,164	63.2	11.5	11.0	6.5	41.3	26.8	20.8	7.3
middle # 5,175	W	3,732	2.6	28.6	10.4	12.5	13.6	5.8	3.7	0.7
	M	3,306	41.3	9.3	6.5	3.1	18.7	13.1	10.4	3.8
	WM	4,392	32.0	9.8	9.3	7.4	20.9	12.3	9.3	3.2
low # 5,117	W	3,051	2.3	11.9	2.1	3.3	2.6	1.2	0.9	0.3
	M	2,178	30.1	2.8	1.2	0.5	2.5	1.5	1.5	0.9
	WM	3,687	18.9	3.5	2.1	2.4	3.3	1.7	1.5	0.7

Table 2: Evaluation of semantic neighbor extraction

sources were filtered to discard entries and synonyms that are not part of the AQUAINT-2 vocabulary (see the difference between the number of words in the first column and the number of evaluated words of the third column). Since the frequency of words is an important factor in distributional approaches, we give our results globally but also for three ranges of frequencies that split our set of nouns into roughly equal parts: *high* frequency (frequency > 1000), *middle* frequency (100 < frequency ≤ 1000) and *low* frequency (10 < frequency ≤ 100). These results take the form of several measures and start at the fifth column by the proportion of the synonyms and similar words of our references that are found among the first 100 extracted neighbors of each noun. As these neighbors are ranked according to their similarity value with their target word, the evaluation measures are taken from the Information Retrieval field by replacing documents with synonyms and queries with target words (see the four last columns of Table 2). The R-precision (R-prec.) is the precision after the first R neighbors were retrieved, R being the number of reference synonyms; the Mean Average Precision (MAP) is the average of the precision value after a reference synonym is found; precision at different cut-offs is given for the 1, 5, 10 and 100 first neighbors. All these values are given as percentages.

The results of Table 2 lead to three main observations. First, the level of results heavily depends on the frequency range of target words: the best results are obtained for high frequency words while evaluation measures significantly decrease for words whose frequency is low. Sec-

ond, the characteristics of the reference resources have a significant impact on results. WordNet provides a restricted number of synonyms for each noun while the Moby thesaurus contains for each entry a large number of synonyms and similar words. As a consequence, the precisions at different cut-offs have a significantly higher value with Moby as reference than with WordNet as reference. Finally, the results of Table 2 are compatible with those of (Lin, 1998) for instance (R-prec. = 11.6 and MAP = 8.1 with WM as reference for all entries of the thesaurus at <http://webdocs.cs.ualberta.ca/lindek/Downloads/sim.tgz>) if we take into account the fact that the thesaurus of Lin was built from a much larger corpus and with syntactic co-occurrences.

## 4.2 Implementation issues

The implementation of the method we have presented in section 3 raises several issues. One of these concerns the occurrences to select from texts of both the entries of the thesaurus and their neighbors. These occurrences are used both for the training of our word-in-context classifier and for the identification of bad neighbors. In practice, we extract randomly from our reference corpus, *i.e.* the AQUAINT-2 corpus, a fixed number of sentences, equal to 250, for each word of the vocabulary of our initial thesaurus and exploit them for the two tasks. This extraction is performed on the basis of the lemma form of these words. It should be noted that 250 is the upper limit of the number of occurrences by word since the frequency in the corpus of many words is lower than 250. When this limit is not reached, all the available oc-

currences are taken, which may be no more than 11 occurrences for certain low-frequency words. The upper limit of 250 is halfway between the 385 training examples on average for the Lexical Sample Task of Senseval 1 and the 118 training examples on average for the same task of Senseval 2.

The training of our word-in-context classifier is also an important issue. As mentioned before, this classifier is a linear SVM. Hence, only its  $C$  regularization parameter can be optimized. Since we have one specific classifier for each thesaurus entry, such optimization has globally a high cost, even for a linear kernel. Hence, we have first evaluated through a 5-fold cross-validation method the results of these classifiers with a default value of  $C$ , equal to 1. Table 3 gives their average accuracy value along with their standard deviation for all the entries of the thesaurus and for the three frequency ranges of Table 2.

	all	high	middle	low
accuracy	86.2	86.1	86.0	86.5
standard deviation	6.1	4.2	5.7	7.6

Table 3: Results of word-in-context classifiers

This table shows a global high level of result along with similar values for all the frequency ranges of entries<sup>4</sup>. Hence, we have decided not to optimize the  $C$  parameter and to adopt the default value of 1 for all the word-in-context classifiers.

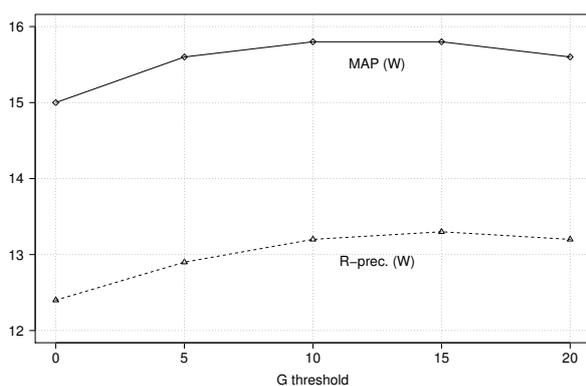


Figure 1: R-precision and MAP for various values of the  $G$  threshold

The last and the most important implementation issue is the setting of the threshold  $G$  for determining whether a neighbor is likely to be a bad

<sup>4</sup>The standard deviation is a little bit higher for the lowest frequencies but it should be noted that the low number of examples for low frequency entries does not seem to have a strong impact on the results of such classifier.

neighbor. For this setting, we have randomly chosen a subset of 859 entries of our initial thesaurus that corresponds to 10% of the entries with at least one true neighbor in any of our references. Figure 1 gives the results of the reranked thesaurus for these entries in terms of R-precision and MAP against reference  $W^5$  for various values of  $G$ . Although the level of these measures does not change a lot for  $G > 5$ , the graph of Figure 1 shows that  $G = 15$  appears to be an optimal value. Hence, this is the value used for the detailed evaluation of the next section.

### 4.3 Evaluation of the reranked thesaurus

Table 4 gives the evaluation of the application of our reranking method to the initial thesaurus according to the same principles as in section 4.1. The value of each measure comes with its difference with the corresponding value for the initial thesaurus. As the recall measure and the precision for the last rank do not change in a reranking process, they are not given again.

The first thing to notice is that at the global scale, all measures for all references are significantly improved<sup>6</sup>, which means that our hypothesis about the possibility for a discriminative classifier to capture the meaning of a word tends to be validated. It is an interesting result since the features upon which this classifier was built were taken from WSD and were not specifically selected for this task. As a consequence, there is probably some room for improvement.

If we go into details, Table 4 clearly shows two main trends. First, the improvement of results is particularly effective for middle frequency entries, then for low frequency and finally, for high frequency entries. Because of their already high level in the initial thesaurus, results for high frequency entries are difficult to improve but it is important to note that our selection of bad neighbors has a very low error rate, which at least preserves these results. This is confirmed by the fact that, with WordNet as reference, only 744 neighbors were found wrongly downgraded, spread over 686 entries, which represents only 5% of all downgraded neighbors. The second main trend of Table 4 con-

<sup>5</sup>The use of  $W$  as reference is justified by the fact that the number of synonyms for an entry in  $W$  is more compatible, especially for R-precision, with the real use of the resulting thesaurus in an application.

<sup>6</sup>The statistical significance of differences with the initial thesaurus was evaluated by a paired Wilcoxon test with  $p$ -value  $< 0.05$  and  $< 0.01$  ( $\dagger$  and  $\ddagger$  for non significance).

freq.	ref.	R-prec.	MAP	P@1	P@5	P@10
all	W	9.1 (0.9)	10.7 (0.9)	12.8 (1.1)	5.6 (0.5)	3.7 (0.3)
	M	7.2 (0.5)	3.5 (0.3)	26.5 (2.4)	17.9 (1.5)	14.0 (1.0)
	WM	8.4 (0.7)	6.1 (0.5)	24.8 (2.3)	15.4 (1.3)	11.7 (0.9)
high	W	11.3 (0.2) †	12.6 (0.1)	17.3 (0.1) ‡	7.8 (0.1) ‡	5.1 (0.0)
	M	10.3 (0.1)	4.9 (0.0)	42.1 (0.8)	28.4 (0.4)	22.1 (0.2)
	WM	11.1 (0.1)	6.6 (0.1)	42.0 (0.7)	27.2 (0.4)	20.9 (0.1)
middle	W	11.8 (1.4)	13.8 (1.3)	15.7 (2.1)	6.5 (0.7)	4.1 (0.4)
	M	7.3 (0.8)	3.6 (0.5)	23.3 (4.6)	16.0 (2.9)	12.4 (2.0)
	WM	10.3 (1.0)	8.1 (0.7)	25.1 (4.2)	14.6 (2.3)	10.9 (1.6)
low	W	3.2 (1.1)	4.6 (1.3)	3.9 (1.3)	1.8 (0.6)	1.3 (0.4)
	M	1.8 (0.6)	0.8 (0.3)	4.4 (1.9)	2.9 (1.4)	2.6 (1.1)
	WM	3.1 (1.0)	3.3 (0.9)	5.1 (1.8)	2.9 (1.2)	2.3 (0.8)

Table 4: Results of the reranking of semantic neighbors

cerns the type of semantic relations: results with Moby as reference are improved in a larger extent than results with WordNet as reference. This suggests that our procedure is more effective for semantically related words than for semantically similar words, which can be considered as a little bit surprising since the notion of context in our discriminative classifier seems *a priori* more strict than in “classical” distributional contexts. However, this point must be investigated further as a significant part of the relations in Moby, even if they do not represent the largest part of them, are paradigmatic relations.

<b>WordNet</b>	respect, admiration, regard
<u>Moby</u>	admiration, appreciation, acceptance, dignity, regard, respect, account, adherence, consideration, estimate, estimation, fame, greatness, reverence + 79 words more
initial	cordiality, gratitude, <b>admiration</b> , comradeship, back-scratching, perplexity, <b>respect</b> , ruination, appreciation, neighbourliness . . .
reranking	gratitude, <b>admiration</b> , <b>respect</b> , appreciation, neighborliness, trust, empathy, goodwill, reciprocity, half-staff, affection, self-esteem, reverence, longing, <b>regard</b> . . .

Table 5: Impact of our reranking for the entry *esteem*

Table 5 illustrates more precisely the impact of our reranking procedure for the middle frequency entry *esteem*. Its **WordNet** row gives all the reference synonyms for this entry in WordNet while its Moby row gives the first reference related words

for this entry in Moby. In our *initial* thesaurus, the first two neighbors of *esteem* that are present in our reference resources are *admiration* (rank 3) and *respect* (rank 7). The reranking produces a thesaurus in which these two words appear as the second and the third neighbors of the entry because neighbors without clear relation with it such as *back-scratching* were downgraded while its third synonym in WordNet is raised from rank 22 to rank 15. Moreover, the number of neighbors among the first 15 ones that are present in Moby increases from 3 to 5.

## 5 Related work

The building of distributional thesaurus is generally viewed as an application or a mode of evaluation of work about semantic similarity or semantic relatedness. As a consequence, the improvement of such thesaurus is generally not directly addressed but is a possible consequence of the improvement of semantic similarity measures. However, the extent of this improvement is rarely evaluated as most of the work about semantic similarity is evaluated on datasets such as the WordSim-353 test collection (Gabrilovich and Markovitch, 2007), which are only partially representative of the results for thesaurus building.

If we consider more specifically the problem of improving semantic similarity, and by the way thesauri, in a given paradigm, (Broda et al., 2009), (Zhitomirsky-Geffet and Dagan, 2009) and (Yamamoto and Asakura, 2010), which all take place in the paradigm defined by (Grefenstette, 1994), are the closest works to ours. (Broda et al., 2009) proposes a new weighting scheme of words in distributional contexts that replaces the weight of

word by a function of its rank in the context, which is a way to be less dependent on the values of a particular weighting function. (Zhitomirsky-Geffet and Dagan, 2009) shares with our work the use of bootstrapping by relying on an initial thesaurus to derive means of improving it. More specifically, (Zhitomirsky-Geffet and Dagan, 2009) assumes that the first neighbors of an entry are more relevant than the others and as a consequence, that their most significant features are also representative of the meaning of the entry. The neighbors of the entry are reranked according to this hypothesis by increasing the weight of these features to favor their influence in the distributional contexts that support the evaluation of the similarity between the entry and its neighbors. (Yamamoto and Asakura, 2010) is a variant of (Zhitomirsky-Geffet and Dagan, 2009) that takes into account a larger number of features for the reranking process. One main difference between all these works and ours is that they assume that the initial thesaurus was built by relying on distributional contexts represented as bags-of-words. Our method does not make this assumption as its reranking is based on a classifier built in an unsupervised way<sup>7</sup> from and applied to the corpus used for building the initial thesaurus. As a consequence, it could even be applied to other paradigms than (Grefenstette, 1994).

If we focus more specifically on the improvement of distributional thesauri, (Ferret, 2012) is the most comparable work to ours, both because it is specifically focused on this task and it is based on the same evaluation framework. (Ferret, 2012) selects in an unsupervised way a set of positive and negative examples of semantically similar words from the initial thesaurus, uses them for training a classifier deciding whether or not a pair of words are semantically similar and finally, applies this classifier to the neighbors of each entry for reranking them. One of the objectives of (Ferret, 2012) was to rebalance the initial thesaurus in favor of low frequency entries. Although this objective was reached, the resulting thesaurus tends to have a lower performance than the initial thesaurus for high frequency entries and for synonyms. The problem with high frequency entries comes from the fact that applying a machine learning classifier to its training examples does not lead to a perfect result. The problem with synonyms

---

<sup>7</sup>It is a supervised classifier but its training set is selected in an unsupervised way.

arises from the imbalance between *semantic similarity* and *semantic relatedness* among training examples: most of selected examples were pairs of words linked by *semantic relatedness* because this kind of relations are more frequent among semantic neighbors than relations based on *semantic similarity*.

In both cases, the method proposed in (Ferret, 2012) faces the problem of relying only on the distributional thesaurus it tries to improve. This is an important difference with the method presented in this article, which mainly exploits the context of the occurrences of words in the corpus used for the building the initial thesaurus. As a consequence, at a global scale, our reranked thesaurus outperforms the final thesaurus of (Ferret, 2012) for nearly all measures. The only exceptions are the P@1 values for M and WM as reference. However, it should be noted that values for both MAP and R-precision, which are more reliable measures than P@1, are identical for the two thesauri and the same references.

## 6 Conclusion and perspectives

In this article, we have presented a new approach for reranking the semantic neighbors of a distributional thesaurus. This approach relies on the unsupervised building of discriminative classifiers dedicated to the identification of its entries in texts, with the objective to characterize their meaning according to the distributional hypothesis. The classifier built for an entry is then applied to a set of occurrences of its neighbors for identifying and downgrading those that are not semantically related to the entry. The proposed method was tested on a large thesaurus of nouns for English and led to a significant improvement of this thesaurus, especially for middle and low frequency entries and for semantic relatedness. We plan to extend this work by taking into account the notion of word sense as it is done in (Reisinger and Mooney, 2010) or (Huang et al., 2012): since we rely on occurrences of words in texts, this extension should be quite straightforward by turning our word-in-context classifiers into true word sense classifiers.

## Acknowledgments

This work was partly supported by the project ANR ASFALDA ANR-12-CORD-0023.

## References

- Andrei Alexandrescu and Katrin Kirchhoff. 2007. Data-driven graph construction for semi-supervised graph-based learning in NLP. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2007)*, pages 204–211, Rochester, New York.
- Satanjeev Bano Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Eighteenth International Conference on Artificial Intelligence (IJCAI-03)*, Acapulco, Mexico.
- Bartosz Broda, Maciej Piasecki, and Stan Szpakowicz. 2009. Rank-Based Transformation in Measuring Semantic Relatedness. In *22<sup>nd</sup> Canadian Conference on Artificial Intelligence*, pages 187–190.
- Alexander Budanitsky and Graeme Hirst. 2006. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.
- James Curran and Marc Moens. 2002a. Scaling context space. In *40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 231–238, Philadelphia, Pennsylvania, USA.
- James R. Curran and Marc Moens. 2002b. Improvements in automatic thesaurus extraction. In *Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, pages 59–66, Philadelphia, USA.
- Katrin Erk and Sebastian Pado. 2010. Exemplar-based models for word meaning in context. In *48<sup>th</sup> Annual Meeting of the Association for Computational Linguistics (ACL 2010), short paper*, pages 92–97, Uppsala, Sweden, July.
- Olivier Ferret. 2010. Testing semantic similarity measures for extracting synonyms from a corpus. In *Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta.
- Olivier Ferret. 2012. Combining bootstrapping and feature selection for improving a distributional thesaurus. In *20<sup>th</sup> European Conference on Artificial Intelligence (ECAI 2012)*, pages 336–341, Montpellier, France.
- John R. Firth, 1957. *Studies in Linguistic Analysis*, chapter A synopsis of linguistic theory 1930-1955, pages 1–32. Blackwell, Oxford.
- Dayne Freitag, Matthias Blume, John Byrnes, Edmond Chow, Sadik Kapadia, Richard Rohwer, and Zhiqiang Wang. 2005. New experiments in distributional representations of synonymy. In *Ninth Conference on Computational Natural Language Learning (CoNLL)*, pages 25–32, Ann Arbor, Michigan, USA.
- Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *20<sup>th</sup> International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 6–12.
- William A Gale, Kenneth W Church, and David Yarowsky. 1992. Work on statistical methods for word sense disambiguation. In *AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pages 54–60.
- Gregory Grefenstette. 1994. *Explorations in automatic thesaurus discovery*. Kluwer Academic Publishers.
- Michael A. K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.
- Kris Heylen, Yves Peirsman, Dirk Geeraerts, and Dirk Speelman. 2008. Modelling Word Similarity: An Evaluation of Automatic Synonymy Extraction Algorithms. In *Sixth conference on International Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco.
- Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *50th Annual Meeting of the Association for Computational Linguistics (ACL'12)*, pages 873–882.
- Jun'ichi Kazama, Stijn De Saeger, Kow Kuroda, Masaki Murata, and Kentaro Torisawa. 2010. A bayesian method for robust estimation of distributional similarities. In *48<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*, pages 247–256, Uppsala, Sweden.
- Yoong Keok Lee and Hwee Tou Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 41–48.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *17<sup>th</sup> International Conference on Computational Linguistics and 36<sup>th</sup> Annual Meeting of the Association for Computational Linguistics (ACL-COLING'98)*, pages 768–774, Montreal, Canada.
- George A. Miller. 1990. WordNet: An On-Line Lexical Database. *International Journal of Lexicography*, 3(4).
- Jane Morris and Graeme Hirst. 2004. Non-classical lexical semantic relations. In *Workshop on Computational Lexical Semantics of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 46–51, Boston, MA.
- Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

- Ted Pedersen, Siddharth Patwardhan, and Jason Mitchell. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *HLT-NAACL 2004, demonstration papers*, pages 38–41, Boston, Massachusetts, USA.
- Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2010)*, pages 109–117, Los Angeles, California, June.
- Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*.
- Grady Ward. 1996. Moby thesaurus. Moby Project.
- Julie Weeds. 2003. *Measures and Applications of Lexical Distributional Similarity*. Ph.D. thesis, Department of Informatics, University of Sussex.
- Kazuhide Yamamoto and Takeshi Asakura. 2010. Even unassociated features can improve lexical distributional similarity. In *Second Workshop on NLP Challenges in the Information Explosion Era (NLPIC 2010)*, pages 32–39, Beijing, China.
- Torsten Zesch and Iryna Gurevych. 2010. Wisdom of crowds versus wisdom of linguists - measuring the semantic relatedness of words. *Natural Language Engineering*, 16(1):25–59.
- Maayan Zhitomirsky-Geffet and Ido Dagan. 2009. Bootstrapping Distributional Feature Vector Quality. *Computational Linguistics*, 35(3):435–461.

# Models of Semantic Representation with Visual Attributes

Carina Silberer, Vittorio Ferrari, Mirella Lapata

School of Informatics, University of Edinburgh

10 Crichton Street, Edinburgh EH8 9AB

c.silberer@ed.ac.uk, vferrari@inf.ed.ac.uk, mlap@inf.ed.ac.uk

## Abstract

We consider the problem of grounding the meaning of words in the physical world and focus on the visual modality which we represent by *visual attributes*. We create a new large-scale taxonomy of visual attributes covering more than 500 concepts and their corresponding 688K images. We use this dataset to train attribute classifiers and integrate their predictions with text-based distributional models of word meaning. We show that these bimodal models give a better fit to human word association data compared to amodal models and word representations based on hand-crafted norming data.

## 1 Introduction

Recent years have seen increased interest in grounded language acquisition, where the goal is to extract representations of the meaning of natural language tied to the physical world. The *language grounding problem* has assumed several guises in the literature such as semantic parsing (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Kate and Mooney, 2007; Lu et al., 2008; Börschinger et al., 2011), mapping natural language instructions to executable actions (Branavan et al., 2009; Tellex et al., 2011), associating simplified language to perceptual data such as images or video (Siskind, 2001; Roy and Pentland, 2002; Gorniak and Roy, 2004; Yu and Ballard, 2007), and learning the meaning of words based on linguistic and perceptual input (Bruni et al., 2012b; Feng and Lapata, 2010; Johns and Jones, 2012; Andrews et al., 2009; Silberer and Lapata, 2012).

In this paper we are concerned with the latter task, namely constructing perceptually grounded

distributional models. The motivation for models that do not learn exclusively from text is twofold. From a cognitive perspective, there is mounting experimental evidence suggesting that our interaction with the physical world plays an important role in the way we process language (Barsalou, 2008; Bornstein et al., 2004; Landau et al., 1998). From an engineering perspective, the ability to learn representations for multimodal data has many practical applications including image retrieval (Datta et al., 2008) and annotation (Chai and Hung, 2008), text illustration (Joshi et al., 2006), object and scene recognition (Lowe, 1999; Oliva and Torralba, 2007; Fei-Fei and Perona, 2005), and robot navigation (Tellex et al., 2011).

One strand of research uses feature norms as a stand-in for sensorimotor experience (Johns and Jones, 2012; Andrews et al., 2009; Steyvers, 2010; Silberer and Lapata, 2012). Feature norms are obtained by asking native speakers to write down attributes they consider important in describing the meaning of a word. The attributes represent perceived physical and functional properties associated with the referents of words. For example, *apples* are typically green or red, round, shiny, smooth, crunchy, tasty, and so on; *dogs* have four legs and bark, whereas *chairs* are used for sitting. Feature norms are instrumental in revealing which dimensions of meaning are psychologically salient, however, their use as a proxy for people's perceptual representations can itself be problematic (Sloman and Ripps, 1998; Zeigenfuse and Lee, 2010). The number and types of attributes generated can vary substantially as a function of the amount of time devoted to each concept. It is not entirely clear how people generate attributes and whether all of these are important for representing concepts. Finally, multiple participants are required to create a representation for each con-

cept, which limits elicitation studies to a small number of concepts and the scope of any computational model based on feature norms.

Another strand of research focuses exclusively on the visual modality, even though the grounding problem could involve auditory, motor, and haptic modalities as well. This is not entirely surprising. Visual input represents a major source of data from which humans can learn semantic representations of linguistic and non-linguistic communicative actions (Regier, 1996). Furthermore, since images are ubiquitous, visual data can be gathered far easier than some of the other modalities. Distributional models that integrate the visual modality have been learned from texts and images (Feng and Lapata, 2010; Bruni et al., 2012b) or from ImageNet (Deng et al., 2009), e.g., by exploiting the fact that images in this database are hierarchically organized according to WordNet synsets (Leong and Mihalcea, 2011). Images are typically represented on the basis of low-level features such as SIFT (Lowe, 2004), whereas texts are treated as bags of words.

Our work also focuses on images as a way of physically grounding the meaning of words. We, however, represent them by high-level *visual attributes* instead of low-level image features. Attributes are not concept or category specific (e.g., animals have stripes and so do clothing items; balls are round, and so are oranges and coins), and thus allow us to express similarities and differences across concepts more easily. Furthermore, attributes allow us to generalize to unseen objects; it is possible to say something about them even though we cannot identify them (e.g., it has a beak and a long tail). We show that this attribute-centric approach to representing images is beneficial for distributional models of lexical meaning. Our attributes are similar to those provided by participants in norming studies, however, importantly they are *learned* from training data (a database of images and their visual attributes) and thus generalize to new images without additional human involvement.

In the following we describe our efforts to create a new large-scale dataset that consists of 688K images that match the same concrete concepts used in the feature norming study of McRae et al. (2005). We derive a taxonomy of 412 visual attributes and explain how we learn attribute classifiers following recent work in computer vision (Lampert et al., 2009; Farhadi et al., 2009). Next,

we show that this attribute-based image representation can be usefully integrated with textual data to create distributional models that give a better fit to human word association data over models that rely on human generated feature norms.

## 2 Related Work

Grounding semantic representations with visual information is an instance of multimodal learning. In this setting the data consists of multiple input modalities with different representations and the learner’s objective is to extract a unified representation that fuses the modalities together. The literature describes several successful approaches to multimodal learning using different variants of deep networks (Ngiam et al., 2011; Srivastava and Salakhutdinov, 2012) and data sources including text, images, audio, and video.

Special-purpose models that address the fusion of distributional meaning with visual information have been also proposed. Feng and Lapata (2010) represent documents and images by a common multimodal vocabulary consisting of textual words and visual terms which they obtain by quantizing SIFT descriptors (Lowe, 2004). Their model is essentially Latent Dirichlet Allocation (LDA, Blei et al., 2003) trained on a corpus of multimodal documents (i.e., BBC news articles and their associated images). Meaning in this model is represented as a vector whose components correspond to word-topic distributions. A related model has been proposed by Bruni et al. (2012b) who obtain distinct representations for the textual and visual modalities. Specifically, they extract a visual space from images contained in the ESP-Game data set (von Ahn and Dabbish, 2004) and a text-based semantic space from a large corpus collection totaling approximately two billion words. They concatenate the two modalities and subsequently project them to a lower-dimensionality space using Singular Value Decomposition (Golub et al., 1981).

Traditionally, computer vision algorithms describe visual phenomena (e.g., objects, scenes, faces, actions) by giving each instance a categorical label (e.g., cat, beer garden, Brad Pitt, drinking). The ability to describe images by their attributes allows to generalize to new instances for which there are no training examples available. Moreover, attributes can transcend category and task boundaries and thus provide a generic description of visual data.

Initial work (Ferrari and Zisserman, 2007)

focused on simple color and texture attributes (e.g., blue, stripes) and showed that these can be learned in a weakly supervised setting from images returned by a search engine when using the attribute as a query. Farhadi et al. (2009) were among the first to use visual attributes in an object recognition task. Using an inventory of 64 attribute labels, they developed a dataset of approximately 12,000 instances representing 20 objects from the PASCAL Visual Object Classes Challenge 2008 (Everingham et al., 2008). Visual semantic attributes (e.g., hairy, four-legged) were used to identify familiar objects and to describe unfamiliar objects when new images and bounding box annotations were provided. Lampert et al. (2009) showed that attribute-based representations can be used to classify objects when there are no training examples of the target classes available. Their dataset contained over 30,000 images representing 50 animal concepts and used 85 attributes from the norming study of Osherson et al. (1991). Attribute-based representations have also been applied to the tasks of face detection (Kumar et al., 2009), action identification (Liu et al., 2011), and scene recognition (Patterson and Hays, 2012).

The use of visual attributes in models of distributional semantics is novel to our knowledge. We argue that they are advantageous for two reasons. Firstly, they are cognitively plausible; humans employ visual attributes when describing the properties of concept classes. Secondly, they occupy the middle ground between non-linguistic low-level image features and linguistic words. Attributes crucially represent image properties, however by being words themselves, they can be easily integrated in any text-based distributional model thus eschewing known difficulties with rendering images into word-like units.

A key prerequisite in describing images by their attributes is the availability of training data for learning attribute classifiers. Although our database shares many features with previous work (Lampert et al., 2009; Farhadi et al., 2009) it differs in focus and scope. Since our goal is to develop distributional models that are applicable to many words, it contains a considerably larger number of concepts (i.e., more than 500) and attributes (i.e., 412) based on a detailed taxonomy which we argue is cognitively plausible and beneficial for image and natural language processing tasks. Our experiments evaluate a number of models previously proposed in the literature and in

Attribute Categories		Example Attributes
color_patterns	(25)	is_red, has_stripes
diet	(35)	eats_nuts, eats_grass
shape_size	(16)	is_small, is_chubby
parts	(125)	has_legs, has_wheels
botany;anatomy	(25;78)	has_seeds, has_fur
behavior (in)animate	(55)	flies, waddles, pecks
texture_material	(36)	made_of_metal, is_shiny
structure	(3)	2_pieces, has_pleats

Table 1: Attribute categories and examples of attribute instances. Parentheses denote the number of attributes per category.

all cases show that the attribute-based representation brings performance improvements over just using the textual modality. Moreover, we show that automatically computed attributes are comparable and in some cases superior to those provided by humans (e.g., in norming studies).

### 3 The Attribute Dataset

**Concepts and Images** We created a dataset of images and their visual attributes for the nouns contained in McRae et al.’s (2005) feature norms. The norms cover a wide range of concrete concepts including animate and inanimate things (e.g., animals, clothing, vehicles, utensils, fruits, and vegetables) and were collected by presenting participants with words and asking them to list properties of the objects to which the words referred. To avoid confusion, in the remainder of this paper we will use the term *attribute* to refer to properties of concepts and the term *feature* to refer to image features, such as color or edges.

Images for the concepts in McRae et al.’s (2005) production norms were harvested from ImageNet (Deng et al., 2009), an ontology of images based on the nominal hierarchy of WordNet (Fellbaum, 1998). ImageNet has more than 14 million images spanning 21K WordNet synsets. We chose this database due to its high coverage and the high quality of its images (i.e., cleanly labeled and high resolution). McRae et al.’s norms contain 541 concepts out of which 516 appear in ImageNet<sup>1</sup> and are represented by 688K images overall. The average number of images per concept is 1,310 with the most popular being *closet* (2,149 images) and the least popular *prune* (5 images).

<sup>1</sup>Some words had to be modified in order to match the correct synset, e.g., *tank\_(container)* was found as *storage\_tank*.

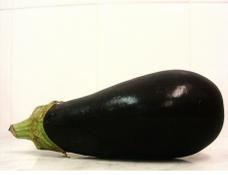
	behavior diet shape_size anatomy  color_patterns	eats, walks, climbs, swims, runs drinks_water, eats_anything is_tall, is_large has_mouth, has_head, has_nose, has_tail, has_claws, has_jaws, has_neck, has_snout, has_feet, has_tongue is_black, is_brown, is_white
	botany color_patterns shape_size texture_material	has_skin, has_seeds, has_stem, has_leaves, has_pulp purple, white, green, has_green_top is_oval, is_long is_shiny
	behavior parts  texture_material color_patterns	rolls has_step_through_frame, has_fork, has_2_wheels, has_chain, has_pedals has_gears, has_handlebar, has_bell, has_breaks, has_seat, has_spokes made_of_metal different_colors, is_black, is_red, is_grey, is_silver

Table 2: Human-authored attributes for *bear*, *eggplant*, and *bike*.

The images depicting each concept were randomly partitioned into a training, development, and test set. For most concepts the development set contained a maximum of 100 images and the test set a maximum of 200 images. Concepts with less than 800 images in total were split into 1/8 test and development set each, and 3/4 training set. The development set was used for devising and refining our attribute annotation scheme. The training and test sets were used for learning and evaluating, respectively, attribute classifiers (see Section 4).

**Attribute Annotation** Our aim was to develop a set of visual attributes that are both discriminating and cognitively plausible, i.e., humans would generally use them to describe a concrete concept. As a starting point, we thus used the visual attributes from McRae et al.’s (2005) norming study. Attributes capturing other primary sensory information (e.g., smell, sound), functional/motor properties, or encyclopaedic information were not taken into account. For example, *is\_purple* is a valid visual attribute for an *eggplant*, whereas *a\_vegetable* is not, since it cannot be visualized. Collating all the visual attributes in the norms resulted in a total of 673 which we further modified and extended during the annotation process explained below.

The annotation was conducted on a *per-concept* rather than a *per-image* basis (as for example in Farhadi et al. (2009)). For each concept (e.g., *bear* or *eggplant*), we inspected the images in the development set and chose all McRae et al. (2005) visual attributes that applied. If an attribute was generally true for the concept, but the images did not

provide enough evidence, the attribute was nevertheless chosen and labeled with `<no_evidence>`. For example, a *plum* has *a\_pit*, but most images in ImageNet show plums where only the outer part of the fruit is visible. Attributes supported by the image data but missing from the norms were added. For example, *has\_lights* and *has\_bumper* are attributes of *cars* but are not included in the norms. Attributes were grouped in eight general classes shown in Table 1. Annotation proceeded on a category-by-category basis, e.g., first all food-related concepts were annotated, then animals, vehicles, and so on. Two annotators (both co-authors of this paper) developed the set of attributes for each category. One annotator first labeled concepts with their attributes, and the other annotator reviewed the annotations, making changes if needed. Annotations were revised and compared per category in order to ensure consistency across all concepts of that category.

Our methodology is slightly different from Lampert et al. (2009) in that we did not simply transfer the attributes from the norms to the concepts in question but refined and extended them according to the visual data. There are several reasons for this. Firstly, it makes sense to select attributes corroborated by the images. Secondly, by looking at the actual images, we could eliminate errors in McRae et al.’s (2005) norms. For example, eight study participants erroneously thought that a *catfish* has *scales*. Thirdly, during the annotation process, we normalized synonymous attributes (e.g., *has\_pit* and *has\_stone*) and attributes that exhibited negligible variations



has\_2\_pieces, has\_pointed\_end, has\_strap, has\_thumb, has\_buckles, has\_heels  
has\_shoe\_laces, has\_soles, is\_black, is\_brown, is\_white, made\_of\_leather, made\_of\_rubber



climbs, climbs\_trees, crawls, hops, jumps, eats, eats\_nuts, is\_small, has\_bushy\_tail  
has\_4\_legs, has\_head, has\_neck, has\_nose, has\_snout, has\_tail, has\_claws  
has\_eyes, has\_feet, has\_toes,



diff\_colours, has\_2\_legs, has\_2\_wheels, has\_windshield, has\_floorboard, has\_stand, has\_tank  
has\_mudguard, has\_seat, has\_exhaust\_pipe, has\_frame, has\_handlebar, has\_lights, has\_mirror  
has\_step-through\_frame, is\_black, is\_blue, is\_red, is\_white, made\_of\_aluminum, made\_of\_steel

Table 3: Attribute predictions for *sandals*, *squirrel*, and *motorcycle*.

in meaning (e.g., *has\_stem* and *has\_stalk*). Finally, our aim was to collect an exhaustive list of visual attributes for each concept which is consistent across all members of a category. This is unfortunately not the case in McRae et al.’s norms. Participants were asked to list up to 14 different properties that describe a concept. As a result, the attributes of a concept denote the set of properties humans consider most salient. For example, both, *lemons* and *oranges* have *has\_pulp*. But the norms provide this attribute only for the second concept.

On average, each concept was annotated with 19 attributes; approximately 14.5 of these were not part of the semantic representation created by McRae et al.’s (2005) participants for that concept even though they figured in the representations of other concepts. Furthermore, on average two McRae et al. attributes per concept were discarded. Examples of concepts and their attributes from our database<sup>2</sup> are shown in Table 2.

#### 4 Attribute-based Classification

Following previous work (Farhadi et al., 2009; Lampert et al., 2009) we learned one classifier per attribute (i.e., 350 classifiers in total).<sup>3</sup> The training set consisted of 91,980 images (with a maximum of 350 images per concept). We used an L2-regularized L2-loss linear SVM (Fan et al., 2008) to learn the attribute predictions. We adopted the training procedure of Farhadi et al. (2009).<sup>4</sup> To learn a classifier for a particular attribute, we used all images in the training data. Images of concepts annotated with the attribute were used as positive examples, and the rest as negative examples. The

data was randomly split into a training and validation set of equal size in order to find the optimal cost parameter  $C$ . The final SVM for the attribute was trained on the entire training data, i.e., on all positive and negative examples.

The SVM learners used the four different feature types proposed in Farhadi et al. (2009), namely color, texture, visual words, and edges. Texture descriptors were computed for each pixel and quantized to the nearest 256 k-means centers. Visual words were constructed with a HOG spatial pyramid. HOG descriptors were quantized into 1000 k-means centers. Edges were detected using a standard Canny detector and their orientations were quantized into eight bins. Color descriptors were sampled for each pixel and quantized to the nearest 128 k-means centers. Shapes and locations were represented by generating histograms for each feature type for each cell in a grid of three vertical and horizontal blocks. Our classifiers used 9,688 features in total. Table 3 shows their predictions for three test images.

Note that attributes are predicted on an image-by-image basis; our task, however, is to describe a concept  $w$  by its visual attributes. Since concepts are represented by many images we must somehow aggregate their attributes into a single representation. For each image  $i_w \in I_w$  of concept  $w$ , we output an  $F$ -dimensional vector containing prediction scores  $\text{score}_a(i_w)$  for attributes  $a = 1, \dots, F$ . We transform these attribute vectors into a single vector  $\mathbf{p}_w \in [0, 1]^{1 \times F}$ , by computing the centroid of all vectors for concept  $w$ . The vector is normalized to obtain a probability distribution over attributes given  $w$ :

$$\mathbf{p}_w = \frac{(\sum_{i_w \in I_w} \text{score}_a(i_w))_{a=1, \dots, F}}{\sum_{a=1}^F \sum_{i_w \in I_w} \text{score}_a(i_w)} \quad (1)$$

We additionally impose a threshold  $\delta$  on  $\mathbf{p}_w$  by set-

<sup>2</sup>Available from <http://homepages.inf.ed.ac.uk/mlap/index.php?page=resources>.

<sup>3</sup>We only trained classifiers for attributes corroborated by the images and excluded those labeled with `<no_evidence>`.

<sup>4</sup><http://vision.cs.uiuc.edu/attributes/>

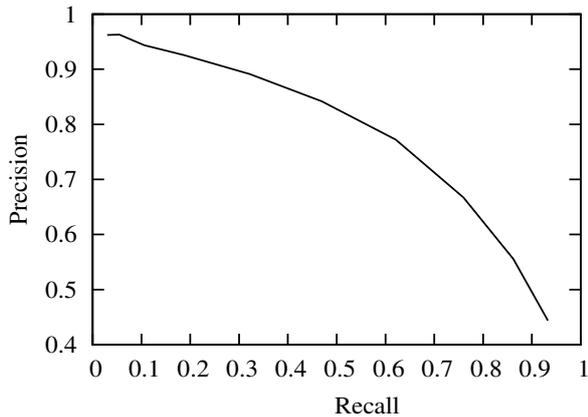


Figure 1: Attribute classifier performance for different thresholds  $\delta$  (test set).

ting each entry less than  $\delta$  to zero.

Figure 1 shows the results of the attribute prediction on the test set on the basis of the computed centroids; specifically, we plot recall against precision based on threshold  $\delta$ .<sup>5</sup> Table 4 shows the 10 nearest neighbors for five example concepts from our dataset. Again, we measure the cosine similarity between a concept and all other concepts in the dataset when these are represented by their visual attribute vector  $\mathbf{p}_w$ .

## 5 Attribute-based Semantic Models

We evaluated the effectiveness of our attribute classifiers by integrating their predictions with traditional text-only models of semantic representation. These models have been previously proposed in the literature and were also described in a recent comparative study (Silberer and Lapata, 2012).

We represent the visual modality by attribute vectors computed as shown in Equation (1). The linguistic environment is approximated by *textual* attributes. We used Strudel (Baroni et al., 2010) to obtain these attributes for the nouns in our dataset. Given a list of target words, Strudel extracts weighted word-attribute pairs from a lemmatized and pos-tagged text corpus (e.g., *eggplant-cook-v*, *eggplant-vegetable-n*). The weight of each word-attribute pair is a log-likelihood ratio score expressing the pair’s strength of association. In our experiments we learned word-attribute pairs from a lemmatized and pos-tagged (2009) dump of the English Wikipedia.<sup>6</sup> In the remainder of this section we will briefly describe the models we

<sup>5</sup>Threshold values ranged from 0 to 0.9 with 0.1 stepsize.

<sup>6</sup>The corpus can be downloaded from <http://wacky.sslmit.unibo.it/doku.php?id=corpora>.

Concept	Nearest Neighbors
boat	ship, sailboat, yacht, submarine, canoe, whale, airplane, jet, helicopter, tank_(army)
rooster	chicken, turkey, owl, pheasant, peacock, stork, pigeon, woodpecker, dove, raven
shirt	blouse, robe, cape, vest, dress, coat, jacket, skirt, camisole, nightgown
spinach	lettuce, parsley, peas, celery, broccoli, cabbage, cucumber, rhubarb, zucchini, asparagus
squirrel	chipmunk, raccoon, groundhog, gopher, porcupine, hare, rabbit, fox, mole, emu

Table 4: Ten most similar concepts computed on the basis of averaged attribute vectors and ordered according to cosine similarity.

used in our study and how the textual and visual modalities were fused to create a joint representation.

**Concatenation Model** Variants of this model were originally proposed in Bruni et al. (2011) and Johns and Jones (2012). Let  $T \in \mathbb{R}^{N \times D}$  denote a term-attribute co-occurrence matrix, where each cell records a weighted co-occurrence score of a word and a textual attribute. Let  $P \in [0, 1]^{N \times F}$  denote a visual matrix, representing a probability distribution over visual attributes for each word. A word’s meaning can be then represented by the concatenation of its normalized textual and visual vectors.

**Canonical Correlation Analysis** The second model uses Canonical Correlation Analysis (CCA, Hardoon et al. (2004)) to learn a joint semantic representation from the textual and visual modalities. Given two random variables  $\mathbf{x}$  and  $\mathbf{y}$  (or two sets of vectors), CCA can be seen as determining two sets of basis vectors in such a way, that the correlation between the projections of the variables onto these bases is mutually maximized (Borga, 2001). In effect, the representation-specific details pertaining to the two views of the same phenomenon are discarded and the underlying hidden factors responsible for the correlation are revealed.

The linguistic and visual views are the same as in the simple concatenation model just explained. We use a kernelized version of CCA (Hardoon et al., 2004) that first projects the data into a higher-dimensional feature space and then performs CCA in this new feature space. The two kernel matrices are  $K_T = TT'$  and  $K_P = PP'$ . After applying CCA we obtain two matrices projected onto  $l$  basis vectors,  $\tilde{T} \in \mathbb{R}^{N \times l}$ , resulting from the projection of the

textual matrix  $T$  onto the new basis and  $\tilde{P} \in \mathbb{R}^{N \times I}$ , resulting from the projection of the corresponding visual attribute matrix. The meaning of a word is then represented by  $\tilde{T}$  or  $\tilde{P}$ .

**Attribute-topic Model** Andrews et al. (2009) present an extension of LDA (Blei et al., 2003) where words in documents and their associated attributes are treated as observed variables that are explained by a generative process. The idea is that each document in a document collection  $\mathcal{D}$  is generated by a mixture of components  $\{x_1, \dots, x_c, \dots, x_C\} \in \mathcal{C}$ , where a component  $x_c$  comprises a latent discourse topic coupled with an attribute cluster. Inducing these attribute-topic components from  $\mathcal{D}$  with the extended LDA model gives two sets of parameters: word probabilities given components  $P_W(w_i|X = x_c)$  for  $w_i$ ,  $i = 1, \dots, n$ , and attribute probabilities given components  $P_A(a_k|X = x_c)$  for  $a_k$ ,  $k = 1, \dots, F$ . For example, most of the probability mass of a component  $x$  would be reserved for the words *shirt*, *coat*, *dress* and the attributes *has\_1\_piece*, *has\_seams*, *made\_of\_material* and so on.

Word meaning in this model is represented by the distribution  $P_{X|W}$  over the learned components. Assuming a uniform distribution over components  $x_c$  in  $\mathcal{D}$ ,  $P_{X|W}$  can be approximated as:

$$P_{X=x_c|W=w_i} = \frac{P(w_i|x_c)P(x_c)}{P(w_i)} \approx \frac{P(w_i|x_c)}{\sum_{l=1}^C P(w_i|x_l)} \quad (2)$$

where  $C$  is the total number of components.

In our work, the training data is a corpus  $\mathcal{D}$  of *textual* attributes (rather than documents). Each attribute is represented as a bag-of-concepts, i.e., words demonstrating the property expressed by the attribute (e.g., *vegetable-n* is a property of *eggplant*, *spinach*, *carrot*). For some of these concepts, our classifiers predict visual attributes. In this case, the concepts are paired with one of their visual attributes. We sample attributes for a concept  $w$  from their distribution given  $w$  (Eq. (1)).

## 6 Experimental Setup

**Evaluation Task** We evaluated the distributional models presented in Section 5 on the word association norms collected by Nelson et al. (1998).<sup>7</sup> These were established by presenting a large number of participants with a cue word (e.g., *rice*) and asking them to name an associate

word in response (e.g., *Chinese*, *wedding*, *food*, *white*). For each cue, the norms provide a set of associates and the frequencies with which they were named. We can thus compute the probability distribution over associates for each cue. Analogously, we can estimate the degree of similarity between a cue and its associates using our models. The norms contain 63,619 unique cue-associate pairs. Of these, 435 pairs were covered by McRae et al. (2005) and our models. We also experimented with 1,716 pairs that were *not* part of McRae et al.’s study but belonged to concepts covered by our attribute taxonomy (e.g., animals, vehicles), and were present in our corpus and ImageNet. Using correlation analysis (Spearman’s  $\rho$ ), we examined the degree of linear relationship between the human cue-associate probabilities and the automatically derived similarity values.<sup>8</sup>

**Parameter Settings** In order to integrate the visual attributes with the models described in Section 5 we must select the appropriate threshold value  $\delta$  (see Eq. (1)). We optimized this value on the development set and obtained best results with  $\delta = 0$ . We also experimented with thresholding the attribute prediction scores and with excluding attributes with low precision. In both cases, we obtained best results when using all attributes. We could apply CCA to the vectors representing each image separately and then compute a weighted centroid on the projected vectors. We refrained from doing this as it involves additional parameters and assumes input different from the other models. We measured the similarity between two words using the cosine of the angle. For the attribute-topic model, the number of predefined components  $C$  was set to 10. In this model, similarity was measured as defined by Griffiths et al. (2007). The underlying idea is that word association can be expressed as a conditional distribution.

With regard to the textual attributes, we obtained a 9,394-dimensional semantic space after discarding word-attribute pairs with a log-likelihood ratio score less than 19.<sup>9</sup> We also discarded attributes co-occurring with less than two different words.

<sup>8</sup>Previous work (Griffiths et al., 2007) which also predicts word association reports how many times the word with the highest score under the model was the first associate in the human norms. This evaluation metric assumes that there are many associates for a given cue which unfortunately is not the case in our study which is restricted to the concepts represented in our attribute taxonomy.

<sup>9</sup>Baroni et al. (2010) use a similar threshold of 19.51.

<sup>7</sup>From <http://w3.usf.edu/FreeAssociation/>.

	Nelson	Concat	CCA	TopicAttr	TextAttr
Concat	0.24				
CCA	0.30	0.72			
TopicAttr	0.26	0.55	0.28		
TextAttr	0.21	0.80	0.83	0.34	
VisAttr	0.23	0.65	0.52	0.40	0.39

Table 5: Correlation matrix for seen Nelson et al. (1998) cue-associate pairs and five distributional models. All correlation coefficients are statistically significant ( $p < 0.01$ ,  $N = 435$ ).

## 7 Results

Our experiments were designed to answer four questions: (1) Do visual attributes improve the performance of distributional models? (2) Are there performance differences among different models, i.e., are some models better suited to the integration of visual information? (3) How do computational models fare against gold standard norming data? (4) Does the attribute-based representation bring advantages over more conventional approaches based on raw image features?

Our results are broken down into seen (Table 5) and unseen (Table 6) concepts. The former are known to the attribute classifiers and form part of our database, whereas the latter are unknown and are not included in McRae et al.’s (2005) norms. We report the correlation coefficients we obtain when human-derived cue-associate probabilities (Nelson et al., 1998) are compared against the simple concatenation model (Concat), CCA, and Andrews et al.’s (2009) attribute-topic model (TopicAttr). We also report the performance of a distributional model that is based solely on the output of our attribute classifiers, i.e., without any textual input (VisAttr) and conversely the performance of a model that uses textual information only (i.e., Strudel attributes) without any visual input (TextAttr). The results are displayed as a correlation matrix so that inter-model correlations can also be observed.

As can be seen in Table 5 (second column), two modalities are in most cases better than one when evaluating model performance on seen data. Differences in correlation coefficients between models with two versus one modality are all statistically significant ( $p < 0.01$  using a  $t$ -test), with the exception of Concat when compared against VisAttr. It is also interesting to note that TopicAttr is the least correlated model when compared against other bimodal models or single modali-

	Nelson	Concat	CCA	TopicAttr	TextAttr
Concat	0.11				
CCA	0.15	0.66			
TopicAttr	0.17	0.69	0.48		
TextAttr	0.11	0.65	0.25	0.39	
VisAttr	0.13	0.57	0.87	0.57	0.34

Table 6: Correlation matrix for unseen Nelson et al. (1998) cue-associate pairs and five distributional models. All correlation coefficients are statistically significant ( $p < 0.01$ ,  $N = 1,716$ ).

ties. This indicates that the latent space obtained by this model is most distinct from its constituent parts (i.e., visual and textual attributes). Perhaps unsurprisingly Concat, CCA, VisAttr, and TextAttr are also highly intercorrelated.

On unseen pairs (see Table 6), Concat fares worse than CCA and TopicAttr, achieving similar performance to TextAttr. CCA and TopicAttr are significantly better than TextAttr and VisAttr ( $p < 0.01$ ). This indicates that our attribute classifiers generalize well beyond the concepts found in our database and can produce useful visual information even on unseen images. Compared to Concat and CCA, TopicAttr obtains a better fit with the human association norms on the unseen data.

To answer our third question, we obtained distributional models from McRae et al.’s (2005) norms and assessed how well they predict Nelson et al.’s (1998) word-associate similarities. Each concept was represented as a vector with dimensions corresponding to attributes generated by participants of the norming study. Vector components were set to the (normalized) frequency with which participants generated the corresponding attribute when presented with the concept. We measured the similarity between two words using the cosine coefficient. Table 7 presents results for different model variants which we created by manipulating the number and type of attributes involved. The first model uses the full set of attributes present in the norms (All Attributes). The second model (Text Attributes) uses all attributes but those classified as visual (e.g., functional, encyclopaedic). The third model (Visual Attributes) considers solely visual attributes.

We observe a similar trend as with our computational models. Taking visual attributes into account increases the fit with Nelson’s (1998) association norms, whereas visual and textual attributes on their own perform worse. Interestingly, CCA’s

Models	Seen
All Attributes	0.28
Text Attributes	0.20
Visual Attributes	0.25

Table 7: Model performance on seen Nelson et al. (1998) cue-associate pairs; models are based on gold human generated attributes (McRae et al., 2005). All correlation coefficients are statistically significant ( $p < 0.01$ ,  $N = 435$ ).

Models	Seen	Unseen
Concat	0.22	0.10
CCA	0.26	0.15
TopicAttr	0.23	0.19
TextAttr	0.20	0.08
VisAttr	0.21	0.13
MixLDA	0.16	0.11

Table 8: Model performance on a subset of Nelson et al. (1998) cue-associate pairs. Seen are concepts known to the attribute classifiers and covered by MixLDA ( $N = 85$ ). Unseen are concepts covered by LDA but unknown to the attribute classifiers ( $N = 388$ ). All correlation coefficients are statistically significant ( $p < 0.05$ ).

performance is comparable to the All Attributes model (see Table 5, second column), despite using automatic attributes (both textual and visual). Furthermore, visual attributes obtained through our classifiers (see Table 5) achieve a marginally lower correlation coefficient against human generated ones (see Table 7).

Finally, to address our last question, we compared our approach against Feng and Lapata (2010) who represent visual information via quantized SIFT features. We trained their MixLDA model on their corpus consisting of 3,361 BBC news documents and corresponding images (Feng and Lapata, 2008). We optimized the model parameters on a development set consisting of cue-associate pairs from Nelson et al. (1998), excluding the concepts in McRae et al. (2005). We used a vocabulary of approximately 6,000 words. The best performing model on the development set used 500 visual terms and 750 topics and the association measure proposed in Griffiths et al. (2007). The test set consisted of 85 seen and 388 unseen cue-associate pairs that were covered by our models and MixLDA.

Table 8 reports correlation coefficients for our models and MixLDA against human probabilities. All attribute-based models significantly outperform MixLDA on seen pairs ( $p < 0.05$  using a  $t$ -test). MixLDA performs on a par with the concatenation model on unseen pairs, however CCA, TopicAttr, and VisAttr are all superior. Although these comparisons should be taken with a grain of salt, given that MixLDA and our models are trained on different corpora (MixLDA assumes that texts and images are collocated, whereas our images do not have collateral text), they seem to indicate that attribute-based information is indeed beneficial.

## 8 Conclusions

In this paper we proposed the use of automatically computed visual attributes as a way of physically grounding word meaning. Our results demonstrate that visual attributes improve the performance of distributional models across the board. On a word association task, CCA and the attribute-topic model give a better fit to human data when compared against simple concatenation and models based on a single modality. CCA consistently outperforms the attribute-topic model on seen data (it is in fact slightly better over a model that uses gold standard human generated attributes), whereas the attribute-topic model generalizes better on unseen data (see Tables 5, 6, and 8). Since the attribute-based representation is general and text-based we argue that it can be conveniently integrated with any type of distributional model or indeed other grounded models that rely on low-level image features (Bruni et al., 2012a; Feng and Lapata, 2010)

In the future, we would like to extend our database to actions and show that this attribute-centric representation is useful for more applied tasks such as image description generation and object recognition. Finally, we have only scratched the surface in terms of possible models for integrating the textual and visual modality. Interesting frameworks which we plan to explore are deep belief networks and Bayesian non-parametrics.

## References

- M. Andrews, G. Vigliocco, and D. Vinson. 2009. Integrating Experiential and Distributional Data to Learn Semantic Representations. *Psychological Review*, 116(3):463–498.
- M. Baroni, B. Murphy, E. Barbu, and M. Poesio. 2010. Strudel: A Corpus-Based Semantic Model

- Based on Properties and Types. *Cognitive Science*, 34(2):222–254.
- L. W. Barsalou. 2008. Grounded Cognition. *Annual Review of Psychology*, 59:617–845.
- D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March.
- M. Borga. 2001. Canonical Correlation – a Tutorial, January.
- M. H. Bornstein, L. R. Cote, S. Maital, K. Painter, S.-Y. Park, L. Pascual, M. G. Pêcheux, J. Ruel, P. Venuti, and A. Vyt. 2004. Cross-linguistic Analysis of Vocabulary in Young Children: Spanish, Dutch, French, Hebrew, Italian, Korean, and American English. *Child Development*, 75(4):1115–1139.
- B. Börschinger, B. K. Jones, and M. Johnson. 2011. Reducing Grounded Learning Tasks to Grammatical Inference. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1416–1425, Edinburgh, UK.
- S.R.K. Branavan, H. Chen, L. S. Zettlemoyer, and R. Barzilay. 2009. Reinforcement Learning for Mapping Instructions to Actions. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 82–90, Suntec, Singapore.
- E. Bruni, G. Tran, and M. Baroni. 2011. Distributional Semantics from Text and Images. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 22–32, Edinburgh, UK.
- E. Bruni, G. Boleda, M. Baroni, and N. Tran. 2012a. Distributional Semantics in Technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145, Jeju Island, Korea.
- E. Bruni, J. Uijlings, M. Baroni, and N. Sebe. 2012b. Distributional semantics with eyes: Using image analysis to improve computational representations of word meaning. In *Proceedings of the 20th ACM International Conference on Multimedia*, pages 1219–1228., New York, NY.
- C. Chai and C. Hung. 2008. Automatically Annotating Images with Keywords: A Review of Image Annotation Systems. *Recent Patents on Computer Science*, 1:55–68.
- R. Datta, D. Joshi, J. Li, and J. Z. Wang. 2008. Image Retrieval: Ideas, Influences, and Trends of the New Age. *ACM Computing Surveys*, 40(2):1–60.
- J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, Florida.
- M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2008. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. <http://www.pascal-network.org/challenges/VOC/voc2008/workshop>.
- R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874.
- A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. 2009. Describing Objects by their Attributes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1778–1785, Miami Beach, Florida.
- L. Fei-Fei and P. Perona. 2005. A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 524–531, San Diego, California.
- C. Fellbaum, editor. 1998. *WordNet: an Electronic Lexical Database*. MIT Press.
- Y. Feng and M. Lapata. 2008. Automatic image annotation using auxiliary text information. In *Proceedings of ACL-08: HLT*, pages 272–280, Columbus, Ohio.
- Y. Feng and M. Lapata. 2010. Visual Information in Semantic Representation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 91–99, Los Angeles, California. ACL.
- V. Ferrari and A. Zisserman. 2007. Learning Visual Attributes. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 433–440. MIT Press, Cambridge, Massachusetts.
- G. H. Golub, F. T. Luk, and M. L. Overton. 1981. A block lanczos method for computing the singular values and corresponding singular vectors of a matrix. *ACM Transactions on Mathematical Software*, 7:149–169.
- P. Gorniak and D. Roy. 2004. Grounded Semantic Composition for Visual Scenes. *Journal of Artificial Intelligence Research*, 21:429–470.
- T. L. Griffiths, M. Steyvers, and J. B. Tenenbaum. 2007. Topics in Semantic Representation. *Psychological Review*, 114(2):211–244.
- D. R. Hardoon, S. R. Szedmak, and J. R. Shawe-Taylor. 2004. Canonical Correlation Analysis: An Overview with Application to Learning Methods. *Neural Computation*, 16(12):2639–2664.
- B. T. Johns and M. N. Jones. 2012. Perceptual Inference through Global Lexical Similarity. *Topics in Cognitive Science*, 4(1):103–120.
- D. Joshi, J.Z. Wang, and J. Li. 2006. The Story Picturing Engine—A System for Automatic Text illustration. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2(1):68–89.

- R. J. Kate and R. J. Mooney. 2007. Learning Language Semantics from Ambiguous Supervision. In *Proceedings of the 22nd Conference on Artificial Intelligence*, pages 895–900, Vancouver, Canada.
- N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. 2009. Attribute and Simile Classifiers for Face Verification. In *Proceedings of the IEEE 12th International Conference on Computer Vision*, pages 365–372, Kyoto, Japan.
- C. H. Lampert, H. Nickisch, and S. Harmeling. 2009. Learning To Detect Unseen Object Classes by Between-Class Attribute Transfer. In *Computer Vision and Pattern Recognition*, pages 951–958, Miami Beach, Florida.
- B. Landau, L. Smith, and S. Jones. 1998. Object Perception and Object Naming in Early Development. *Trends in Cognitive Science*, 27:19–24.
- C. Leong and R. Mihalcea. 2011. Going Beyond Text: A Hybrid Image-Text Approach for Measuring Word Relatedness. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1403–1407, Chiang Mai, Thailand.
- J. Liu, B. Kuipers, and S. Savarese. 2011. Recognizing Human Actions by Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3337–3344, Colorado Springs, Colorado.
- D. G. Lowe. 1999. Object Recognition from Local Scale-invariant Features. In *Proceedings of the International Conference on Computer Vision*, pages 1150–1157, Corfu, Greece.
- D. Lowe. 2004. Distinctive Image Features from Scale-invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110.
- W. Lu, H. T. Ng, W.S. Lee, and L. S. Zettlemoyer. 2008. A Generative Model for Parsing Natural Language to Meaning Representations. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 783–792, Honolulu, Hawaii.
- K. McRae, G. S. Cree, M. S. Seidenberg, and C. McNorgan. 2005. Semantic Feature Production Norms for a Large Set of Living and Nonliving Things. *Behavior Research Methods*, 37(4):547–559.
- D. L. Nelson, C. L. McEvoy, and T. A. Schreiber. 1998. The University of South Florida Word Association, Rhyme, and Word Fragment Norms.
- J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. 2011. Multimodal deep learning. In *Proceedings of the 28th International Conference on Machine Learning*, pages 689–696, Bellevue, Washington.
- A. Oliva and A. Torralba. 2007. The Role of Context in Object Recognition. *Trends in Cognitive Sciences*, 11(12):520–527.
- D. N. Osherson, J. Stern, O. Wilkie, M. Stob, and E. E. Smith. 1991. Default Probability. *Cognitive Science*, 2(15):251–269.
- G. Patterson and J. Hays. 2012. SUN Attribute Database: Discovering, Annotating and Recognizing Scene Attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2751–2758, Providence, Rhode Island.
- T. Regier. 1996. *The Human Semantic Potential*. MIT Press, Cambridge, Massachusetts.
- D. Roy and A. Pentland. 2002. Learning Words from Sights and Sounds: A Computational Model. *Cognitive Science*, 26(1):113–146.
- C. Silberer and M. Lapata. 2012. Grounded Models of Semantic Representation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1423–1433, Jeju Island, Korea.
- J. M. Siskind. 2001. Grounding the Lexical Semantics of Verbs in Visual Perception using Force Dynamics and Event Logic. *Journal of Artificial Intelligence Research*, 15:31–90.
- S. A. Sloman and L. J. Ripps. 1998. Similarity as an Explanatory Construct. *Cognition*, 65:87–101.
- N. Srivastava and R. Salakhutdinov. 2012. Multimodal learning with deep boltzmann machines. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems*, pages 2231–2239, Lake Tahoe, Nevada.
- M. Steyvers. 2010. Combining feature norms and text data with topic models. *Acta Psychologica*, 133(3):234–342.
- S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. Gopal Banerjee, S. Teller, and N. Roy. 2011. Understanding Natural Language Commands for Robotic Navigation and Manipulation. In *Proceedings of the 25th National Conference on Artificial Intelligence*, pages 1507–1514, San Francisco, California.
- L. von Ahn and L. Dabbish. 2004. Labeling images with a computer game. In *Proceedings of the Human Factors in Computing Systems Conference*, pages 319–326, Vienna, Austria.
- C. Yu and D. H. Ballard. 2007. A Unified Model of Early Word Learning Integrating Statistical and Social Cues. *Neurocomputing*, 70:2149–2165.
- M. D. Zeigenfuse and M. D. Lee. 2010. Finding the Features that Represent Stimuli. *Acta Psychologica*, 133(3):283–295.
- J. M. Zelle and R. J. Mooney. 1996. Learning to Parse Database Queries Using Inductive Logic Programming. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 1050–1055, Portland, Oregon.
- L. S. Zettlemoyer and M. Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorical Grammars. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 658–666, Edinburgh, UK.

# Real-World Semi-Supervised Learning of POS-Taggers for Low-Resource Languages

Dan Garrette<sup>1</sup>

Jason Mielens<sup>2</sup>

Jason Baldridge<sup>2</sup>

<sup>1</sup>Department of Computer Science  
The University of Texas at Austin  
dhg@cs.utexas.edu

<sup>2</sup>Department of Linguistics  
The University of Texas at Austin  
{jmielens, jbaldrid}@utexas.edu

## Abstract

Developing natural language processing tools for low-resource languages often requires creating resources from scratch. While a variety of semi-supervised methods exist for training from incomplete data, there are open questions regarding what types of training data should be used and how much is necessary. We discuss a series of experiments designed to shed light on such questions in the context of part-of-speech tagging. We obtain timed annotations from linguists for the low-resource languages Kinyarwanda and Malagasy (as well as English) and evaluate how the amounts of various kinds of data affect performance of a trained POS-tagger. Our results show that annotation of word types is the most important, provided a sufficiently capable semi-supervised learning infrastructure is in place to project type information onto a raw corpus. We also show that finite-state morphological analyzers are effective sources of type information when few labeled examples are available.

## 1 Introduction

Low-resource languages present a particularly difficult challenge for natural language processing tasks. For example, supervised learning methods can provide high accuracy for part-of-speech (POS) tagging (Manning, 2011), but they perform poorly when little supervision is available. Good results in weakly-supervised tagging have been obtained by training sequence models such as hidden Markov models (HMM) using the Expectation-Maximization algorithm (EM), however most work in this area has still relied on relatively large amounts of data, both annotated and

unannotated, as well as an assumption that the annotations are very clean (Kupiec, 1992; Merialdo, 1994).

The ability to learn taggers using very little data is enticing: only a tiny fraction of the world's languages have enough data for standard supervised models to work well. The collection or development of resources is a time-consuming and expensive process, creating a significant barrier for an under-studied language where there are few experts and little funding. It is thus important to develop approaches that achieve good accuracy based on the amount of data that can be reasonably obtained, for example, in just a few hours by a linguist doing fieldwork on a non-native language.

Previous work explored learning taggers from weak information, but the type, amount, quality, and sources of data raise questions about the applicability of those results to real-world low-resource scenarios (Toutanova and Johnson, 2008; Ravi and Knight, 2009; Hasan and Ng, 2009; Garrette and Baldridge, 2012). Most research simulated weak supervision with tag dictionaries extracted from existing large, expertly-annotated corpora. These resources have been developed over long periods of time by trained annotators who collaborate to produce high-quality analyses. They are also biased towards including only the most likely tag for each word type, resulting in a cleaner dictionary than one would find in a real scenario. As such, these experiments do not reflect real-world constraints.

One exception to this work is Goldberg et al. (2008): they use a manually-constructed lexicon for Hebrew in order to learn an HMM tagger. However, this lexicon was constructed by trained lexicographers over a long period of time and achieves very high coverage of the language with very good quality, much better than could be achieved by our non-expert linguistics graduate student annotators in just a few hours. Cucerzan and Yarowsky

(2002) learn a POS-tagger from existing linguistic resources, namely a dictionary and a reference grammar, but these resources are not available, much less digitized, for most under-studied languages. Haghighi and Klein (2006) develop a model in which a POS-tagger is learned from a list of POS tags and just three “prototype” word types for each tag, but their approach requires a vector space to compute the distributional similarity between prototypes and other word types in the corpus. Such distributional models are not feasible for low-resource languages because they require immense amounts of raw text, much more than is available in these settings (Abney and Bird, 2010). Further, they extracted their prototype lists directly from a labeled corpus, something we are specifically avoiding. Täckström et al. (2013) evaluate the use of mixed type and token constraints generated by projecting information from a high-resource language to a low-resource language via a parallel corpus. However, large parallel corpora are not available for most low-resource languages. These are also expensive resources to create and would take considerably more effort to produce than the monolingual resources that our annotators were able to generate in a four-hour timeframe. Of course, if they are available, such parallel text links could be incorporated into our approach.

In our previous work, we developed a different strategy based on generalizing linguistic input with a computational model: linguists annotated either types or tokens for two hours, these annotations are projected onto a corpus of unlabeled tokens using label propagation and HMMs, and a final POS-tagger is trained on this larger auto-labeled corpus (Garrette and Baldridge, 2013). That approach uses much more realistic types and quantities of resources than previous work; nonetheless, it leaves many open questions regarding the effectiveness of incrementally more annotation, the role of unannotated data, and whether there is a good balance to be found using a *combination* of type- and token-supervision. We also did not consider morphological analyzers as a form of type supervision, as suggested by Merialdo (1994).

This paper addresses these questions via a series of experiments designed to quantify the effect on performance given by the amount of time spent finding or annotating training materials. We specifically look at the impact of four types of data

collection:

1. Time annotating sentences (token supervision)
2. Time creating tag dictionary (type supervision)
3. Time constructing a finite state transducer (FST) to analyze word-type morphology
4. Amount of raw data available for training

We explore these strategies in the context of POS-tagging for Kinyarwanda and Malagasy. We also include experiments for English, pretending as though it is a low-resource language. The overwhelming take away from our results is that type supervision—when backed by an effective semi-supervised learning approach—is the most important source of linguistic information. Also, morphological analyzers help for morphologically rich languages when there are few labeled types or tokens (and, it never hurts to use them). Finally, performance improves with more raw data, though we see diminishing returns past 400,000 tokens. With just four hours of type annotation, our system obtains good accuracy across the three languages: 89.8% on English, 81.9% on Kinyarwanda, and 81.2% on Malagasy.

Our results compare favorably with previous work despite using considerably less supervision and a more difficult set of tags. For example, Li et al. (2012) use the entirety of English Wiktionary directly as a tag dictionary to obtain 87.1% accuracy on English, below our result. Täckström et al. (2013) average 88.8% across 8 major languages, but for Turkish, a morphologically rich language, they achieve only 65.2%, significantly below our 81.9% for morphologically-rich Kinyarwanda.

## 2 Data

Kinyarwanda (KIN) and Malagasy (MLG) are low-resource, KIN is morphologically rich, and English (ENG) is used for comparison. For each language, sentences were divided into four sets: training data to be labeled by annotators, raw training data, development data, and test data.

**Data sources** The KIN texts are transcripts of testimonies by survivors of the Rwandan genocide provided by the Kigali Genocide Memorial Center. The MLG texts are articles from the websites<sup>1</sup> *Lakroa* and *La Gazette* and Malagasy Global Voices.<sup>2</sup> Texts in both KIN and MLG were tok-

<sup>1</sup>[www.lakroa.mg](http://www.lakroa.mg) and [www.lagazette-dgi.com](http://www.lagazette-dgi.com)

<sup>2</sup>[mg.globalvoicesonline.org/](http://mg.globalvoicesonline.org/)

	KIN		MLG		ENG - Experienced		ENG - Novice	
time	type	token	type	token	type	token	type	token
1:00	801	559 (1093)	660	422 (899)	910	522 (1124)	210	308 (599)
2:00	1814	948 (2093)	1363	785 (1923)	2660	1036 (2375)	631	646 (1429)
3:00	2539	1324 (3176)	2043	1082 (3064)	4561	1314 (3222)	1350	953 (2178)
4:00	3682	1651 (4119)	2773	1378 (4227)	6598	1697 (4376)	2185	1220 (2933)

Table 1: Annotations for each language and annotator as time increases. Shows the number of tag dictionary entries from type annotation vs. token. (The count of labeled tokens is shown in parentheses). For brevity, the table only shows hourly progress.

enized and labeled with POS tags by two linguistics graduate students, each of which was studying one of the languages. The KIN and MLG data have 12 and 23 distinct POS tags, respectively.

The Penn Treebank (PTB) (Marcus et al., 1993) is used as ENG data. Section 01 was used for token-supervised annotation, sections 02-14 were used as raw data, 15-18 for development of the FST, 19-21 as a dev set and 22-24 as a test set. The PTB uses 45 distinct POS tags.

**Collecting annotations** Linguists with non-native knowledge of KIN and MLG produced annotations for four hours (in 30-minute intervals) for two tasks. In the first task, type-supervision, the annotator was given a list of the words in the target language (ranked from most to least frequent), and they annotated each word type with its potential POS tags. The word types and frequencies used for this task were taken from the raw training data and did not include the test sets. In the second task, token-supervision, full sentences were annotated with POS tags. The 30-minute intervals allow us to investigate the incremental benefit of additional annotation of each type as well as how both annotation types might be combined within a fixed annotation budget.

Baldrige and Palmer (2009) found that annotator expertise greatly influences effectiveness of active learning for morpheme glossing, a related task. To see how differences in annotator speed and quality impact our task, we obtained ENG data from an *experienced* annotator and a *novice* one.

Ngai and Yarowsky (2000) investigated the effectiveness of rule-writing versus annotation (using active learning) for chunking, and found the latter to be far more effective. While we do not explore a rule-writing approach to POS-tagging, we do consider the impact of rule-based morphological analyzers as a component in our semi-supervised POS-tagging system.

	ENG - Exp.		ENG - Nov.	
time	type	tok	type	tok
1:00	0.05	0.03	0.01	0.02
2:00	0.15	0.05	0.03	0.03
3:00	0.24	0.06	0.07	0.05
4:00	0.32	0.08	0.11	0.06

Table 2: Tag dictionary recall against the test set for ENG annotators on type and token annotations.

**Annotations** Table 1 gives statistics for all languages and annotators showing progress during the 4-hour tasks. With token-annotation, tag dictionary growth slows because high-frequency words are repeatedly annotated, producing only additional frequency and sequence information. In contrast, every type-annotation label is a new tag dictionary entry. For types, growth increases over time, reflecting the fact that high-frequency words (which are addressed first) tend to be more ambiguous and thus require more careful thought than later words. For ENG, we can compare the tagging speed of the experienced annotator with the novice: 50% more tokens and 3 times as many types. The token-tagging speed stayed fairly constant for the experienced annotator, but the novice increased his rate, showing the result of practice.

Checking the annotators' output against the gold tags in the PTB shows that both had good tagging accuracy on tokens: 94-95%. Comparing the tag dictionary entries versus the test data, precision starts in the high 80%*s* and falls to the mid-70%*s* in all cases. However, the differences in recall, shown in Table 2, are more interesting. On types, the experienced annotator maxed out at 32%, but the novice only reaches 11%. Moreover, the maximum for token annotations is much lower due to high repeat-annotation. The discrepancies between experienced and novice, and between type and token recall explain a great deal of the performance disparity seen in the experiments.

### 3 Morphological Transducers

Finite-state transducers (FSTs) accept regular languages and can be constructed easily using regular expressions, which makes them quite useful for phonology, morphology and limited areas of syntax (Karttunen, 2001). Past work has used FSTs for direct POS-tagging (Roche and Schabes, 1995), but this requires tight coupling between the FST and target tagset. We use FSTs for morphological analysis: the FST accepts a word type and produces a set of morphological features. If there are multiple possible analyses for a given word type, the FST returns them all. For instance the Kinyarwanda verb *sibatarazuka* “he is not yet resurrected” is analyzed in several ways:

- +NEG+CL2+IPL+V+arazuk+IMP
- +NEG+CL2+NOT.YET+PRES+zuk+IMP
- +NEG+CL2+NOT.YET+razuk+IMP

FSTs are particularly valuable for their ability to analyze out-of-vocabulary items. By looking for known affixes, FSTs can guess the stem of a word and produce an analysis despite not having knowledge of that stem. For morphologically complex languages like KIN, this ability is especially useful. Other factors, such as a large number of morphologically-conditioned phonological changes (seen in MLG) make out-of-vocabulary guessing more challenging because of the large number of potential stems (high ambiguity).

Development of the FSTs for all three languages was done by iteratively adding rules and lexical items with the goal of increasing coverage on a raw dataset. To accomplish this on a fixed time budget, the most frequently occurring unanalyzed tokens were examined, and their stems plus any observable morphological or phonological patterns were added to the transducer. Additionally, developers searched for known morphological alternations to locate instances of phonological change for inclusion. Coverage was checked against a raw dataset which did not include the test data used for the POS experiments.

The KIN and MLG FSTs were created by English-speaking linguists who were familiar with their respective language. They also used dictionaries and grammars. Each FST was developed in 10 hours. To evaluate the benefits of more development time, a version of the English FST was saved every 30 minutes, as shown in Table 3.

elapsed time	tokens		types	
	count	pct	count	pct
2:00	130k	61%	2.1k	12%
4:00	159k	75%	4.1k	24%
6:00	170k	80%	6.7k	39%
8:00	182k	86%	7.7k	44%
10:00	192k	91%	10.7k	62%

Table 3: Coverage of the English morphological FST during development. For brevity, showing 2-hour increments instead of 30-minute segments.

	tokens		types	
	cov.	ambig.	cov.	ambig.
KIN	86%	2.62	82%	5.31
MLG	78%	2.98	37%	1.13
ENG	91%	1.19	62%	1.97

Table 4: Coverage and ambiguity of the final FST for each language.

### 4 Approach

Learning under low-resource conditions is more difficult than scenarios in most previous POS work because the vast majority of the word types in the training and test data are not covered by the annotations. When most words are unknown, learning algorithms such as EM struggle (Garrette and Baldrige, 2012). Recall that most work on learning POS-taggers from tag dictionaries used tag dictionaries culled from *test* sets (even when considering incomplete dictionaries). We thus build on our previous approach, which exploits extremely sparse, human-generated annotations that are produced without knowledge of which words appear in the test set (Garrette and Baldrige, 2013).

This approach generalizes a small initial tag dictionary to include unannotated word types appearing in raw data. It estimates word/tag pair and tag-transition frequency information using model-minimization, which also reduces noise introduced by automatic tag dictionary expansion. The approach exploits type annotations effectively to learn parameters for out-of-vocabulary words and infer missing frequency and sequence information. This pipeline is described in detail in the previous work, so we give only a brief overview and describe our additions.

The purpose of tag dictionary expansion is to estimate label distributions for tokens in a raw cor-

pus, including words missing in the annotations. For this, a graph connecting annotated words to unannotated words via features is constructed and POS labels are pushed between these items using label propagation (LP) (Talukdar and Crammer, 2009). LP has been used successfully for extending POS labels from high-resource languages to low via parallel corpora (Das and Petrov, 2011; Täckström et al., 2013; Ding, 2011) or high- to low-resource domains (Subramanya et al., 2010), among other tasks. These works have typically used n-gram features (capturing basic syntax) and character affixes (basic morphology).

The character n-gram affix-as-morphology approach produces many features, but only a fraction of them represent actual morphemes. Incorrect features end up pushing noise around the graph, so affixes can lead to more false labels that drown out the true labels. While affixes may be sufficient for languages with limited morphology, their effectiveness diminishes for morphology-rich languages, which have much higher type-to-token ratios. More types means sparser word frequency statistics and more out-of-vocabulary items, and thus problems for EM. Here, we modify the LP graph by supplementing or replacing generic affix features with a focused set of morphological features produced by an FST. These targeted morphological features are effective during LP because words that share them are much more likely to actually share POS tags.

FSTs produce multiple analyses, which is actually advantageous for LP. Ambiguities need not be resolved since we just take the union of all morphological features for all analyses and use them as features in the graph. Note that each FST produces its own POS-tags as features, but these do *not* correspond to the target POS tagset used by the tagger. This is important because it decouples FST development and the final POS task. Thus, any FST for the language, regardless of its provenance, can be used with any target POS tagset.

Since the LP graph contains a node for each corpus token, and each node is labeled with a distribution over POS tags, the graph provides a corpus of sentences labeled with noisy tag *distributions* along with an expanded tag dictionary. This output is useful as input to EM because it contains labels for all seen word types as well as sequence and frequency information. There is a high degree of noise in the LP output, so we employ the model

minimization strategy of Ravi et al. (2010), which finds a minimal set of tag bigrams needed to explain the sentences in the raw corpus. It outputs a corpus of tagged sentences, which are used as a good starting point for EM training of an HMM. The expanded tag dictionary constrains the EM search space by providing a limited tagset for each word type, steering EM towards a desirable result.

Because the HMM trained by EM will contain zero-probabilities for words that did not appear in the training corpus, we use the “auto-supervision” step from our previous work: a Maximum Entropy Markov Model tagger is trained on a corpus that is noisily labeled by the HMM (Garrette and Baldrige, 2012). While training an HMM before the MEMM is not strictly necessary, our tests have shown that this generative-then-discriminative combination generally results in around 3% accuracy improvement.

## 5 Experiments<sup>3</sup>

To better understand the effect that each type of supervision has on tagger accuracy, we perform a series of experiments, with KIN and MLG as true low-resource languages. English experiments, for which we had both experienced and novice annotators, allow for further exploration into issues concerning data collection and preparation.

The overall best accuracies achieved by language are 81.9% for KIN using all types, 81.2% for MLG using half types and half tokens, and 89.8% for ENG using all types and the maximal amount of raw data. All of these best values were achieved using both FST and affix LP features.

All results described in this section are averaged over five folds of raw data.

### 5.1 Types versus tokens

Our primary question was the relationship between annotation type and time. Annotation must be done by someone familiar with the target language, linguistics, and the target POS tagset. For many low-resource languages, such people, and the time they have to spend, are likely to be in short supply. To make the best use of their time, we need to know which annotations are most use-

<sup>3</sup>Code and all MLG data available at [github.com/dhgarrette/low-resource-pos-tagging-2013](https://github.com/dhgarrette/low-resource-pos-tagging-2013). We are unable to provide the KIN or ENG data for download due to licensing restrictions. However, ENG data may be shared with those holding a license for the Penn Treebank and KIN data may be shared on a case-by-case basis.

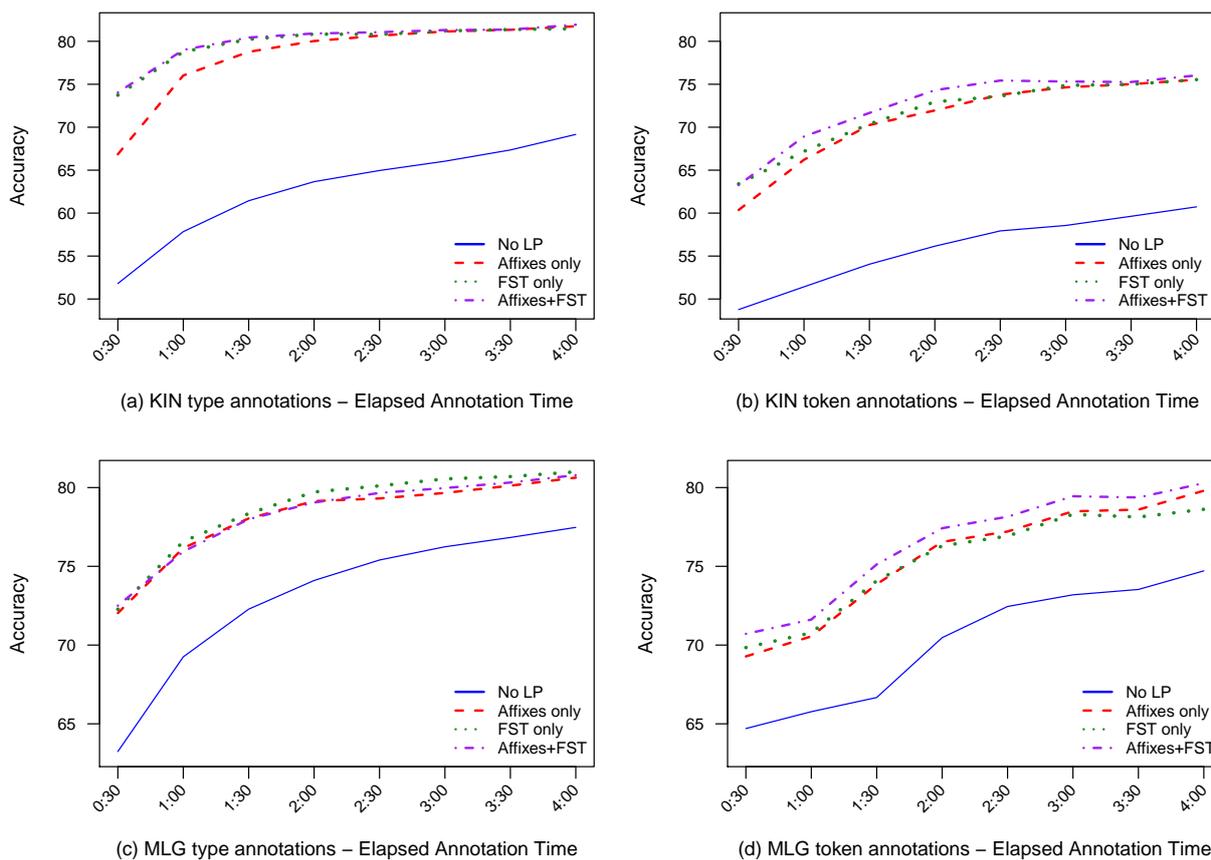


Figure 1: Annotation time vs. tagger accuracy for type-only and token-only annotations.

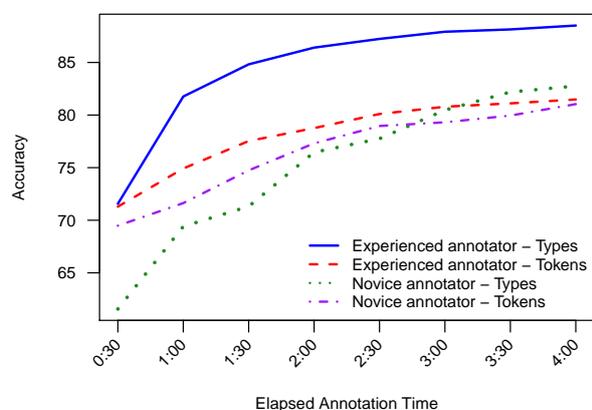


Figure 2: Annotation time vs. tagger accuracy for ENG type-only and token-only annotations with affix and FST LP features.

ful so that efforts can be concentrated there. Additionally, it is useful to identify when returns on annotation effort diminish so that annotators do not spend time doing work that is unlikely to add much value.

The annotators produced four hours each of type and token annotations, each in 30-minute increments. To assess the effects of annotation time,

we trained taggers cumulatively on each increment and determine the value of each additional half-hour of effort. Results are shown for KIN and MLG in Figure 1 and ENG in Figure 2. In all scenarios, the use of LP (and model minimization) delivers huge performance gains. Additionally, the use of FST features, usually along with affixes, yielded better results than without. This indicates the LP procedure makes effective use of the morphological features produced by the FST and that the affix features are able to capture missing information without adding too much noise to the LP graph.

Furthermore, performance is considerably better when type annotations are used than only tokens. Type annotations plateau much faster, so a shorter amount of time must be spent annotating types than if token annotations are used. For KIN it takes approximately 1.5 hours to reach near-maximum accuracy for types, but 2.5 hours for tokens. This difference is due to the fact that the type annotations started with the most frequent words whereas the token annotations were on random sentences. Thus, type annotations quickly cover a significant portion of the language’s tokens. With annotations directly on tokens, some of the highest

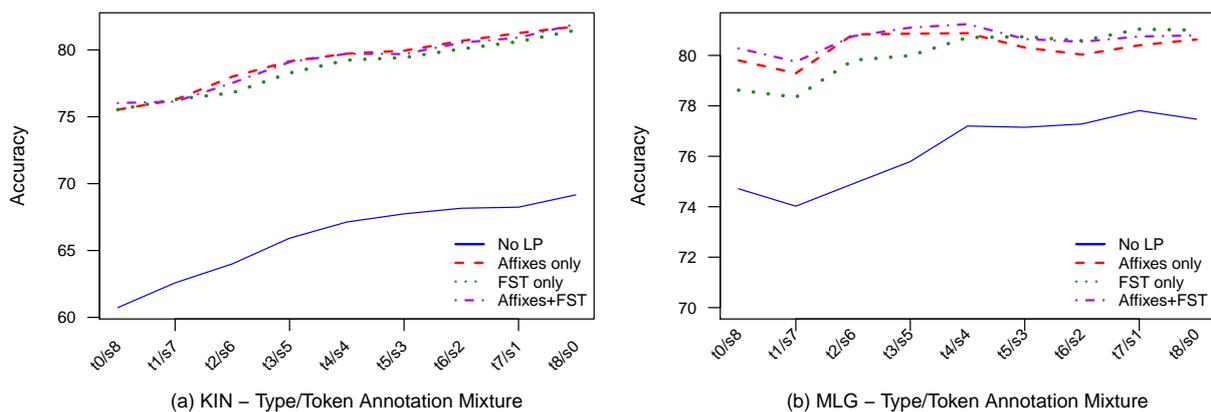


Figure 3: Annotation mixture vs. tagger accuracy. X-axis labels give annotation proportions, e.g. “t2/s6” indicates 2/8 of the time (1 hour) was spent annotating types and 6/8 (3 hours), full sentences.

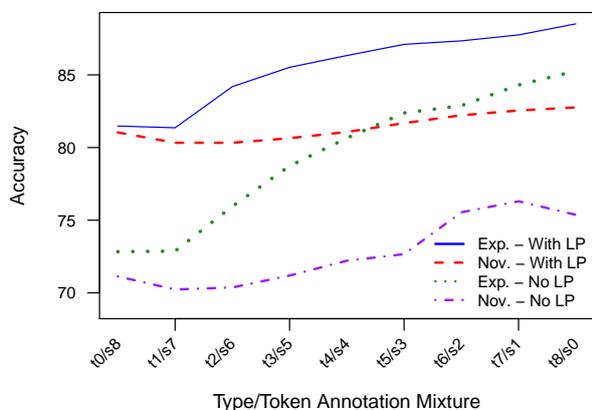


Figure 4: Annotation mixture vs. tagger accuracy on ENG using affix and FST LP features for experienced (Exp.) and novice (Nov.) annotators.

frequency types are covered, but annotation time is also ineffectively used on low-frequency types that happen to appear in those sentences.

Finally, the use of FST features yields the largest gains for KIN, but only when small amounts of annotation are available. This makes sense: KIN is a morphologically rich language, so sparsity is greater and crude affixes capture less actual morphology. With little annotated data, LP relies heavily on morphological features to make clean links between words. But, with more annotations, the gains of the FST over affix features alone diminishes: the affix features eventually capture enough of the morphology to make up the difference.

Figure 2 shows the dramatic differences between the experienced and novice ENG annotators.<sup>4</sup> For the former, results using types and to-

<sup>4</sup>The ENG graph omits “No LP” results since they followed patterns similar to KIN and MLG. Additionally, the results without FST features are not shown because they were nearly identical (though slightly lower) than with the FST.

kens were similar after 30 minutes, but type annotations proved much more useful beyond that. In contrast, the novice annotated types much more slowly, so early on there were not enough annotated types for the training to be as effective. Even so, after three hours of annotation, type annotations still win with the novice, and even beat the experienced annotator labeling tokens.

## 5.2 Mixing type and token annotations

Because type and token annotations are each better at providing different information — a tag dictionary of high-frequency words vs. sequence and frequency information — it is reasonable to expect that a combination of the two might yield higher performance by each contributing different but complementary information during training. This matters in low-resource settings because type or token annotations will likely be produced by the same people, so there is a tradeoff between spending resources on one form of annotation over the other. Understanding the best mixture of annotations can inform us on how to maximize the benefit of a set annotation budget. To this end, we ran experiments fixing the annotation time to four hours while varying the mix of type and token annotations. Results are shown for KIN and MLG in Figure 3 and ENG in Figure 4.

For KIN and ENG, tagger accuracy increases as the proportion of type annotations increases for all LP feature configurations. For MLG, however, as the reliance on the FST increases, the optimal mixture shifts toward higher type proportions. When only affix features are used, the optimal mixture is 1 hour of types and 3 hours of tokens. When FST and affix features are used, the optimum is 2 hours

each of types and tokens. When only FST features are used, it is best to use 3.5 hours of types and only 30 minutes of tokens. Because the FST operates on word types, it is effective at exploiting type annotations. Thus, when the LP focuses more on FST features, it becomes more desirable to have larger amounts of type annotations.

Types clearly win for ENG. The experienced annotator was much faster at annotating types and the speed difference was less pronounced for tokens, so accuracy is most similar when only token annotations are used. The performance disparity grows with increasing the type proportion.

Täckström et al. (2013) explore the use of mixed type and token annotations in which a tagger is learned by projecting information via parallel text. In their experiments, they—like us—found that type information is more valuable than token information. However, they were able to see gains through the complementary effects of mixing type and token annotations. It is likely that this difference in our results is due to the amount of annotated data used. It seems that the amount of type information collected in four hours is not sufficient to saturate the system, meaning that switching to annotating tokens tends to hurt performance.

### 5.3 FST development

The third set of experiments evaluate how the amount of time spent developing an FST affects the performance of trained tagger. To do this, we had our ENG FST developer save progress after each hour (for ten hours). The results show that, for ENG, the FST provided no value, regardless of how much time was spent on its development. Moreover, since large gains in accuracy can be achieved by spending a small amount of time just annotating word types with POS tags, we are led to conclude that time should be spent annotating types or tokens instead of developing an FST. While it is likely that FST development time would have a greater impact for morphologically rich languages, we suspect that greater gains can still be obtained by instead annotating types. Nonetheless, FSTs never seems to *hurt* performance, so if one is readily available, it should be used.

### 5.4 The effect of more raw data

In addition to annotations, semi-supervised tagger training requires a corpus of raw text. Raw data can be easier to acquire since it does not need the attention of a linguist. Even so, for many

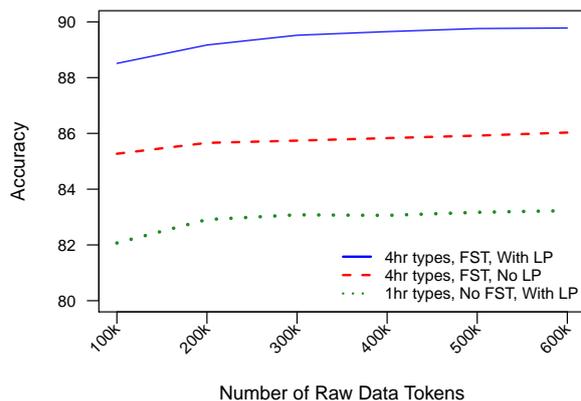


Figure 5: Amount of raw data vs. tagger accuracy for ENG using high vs. low amounts of annotation and using LP vs. no LP., for experienced annotator (novice results were similar).

low-resource languages, the amount of digitized text, such as transcripts or websites, is very limited and may, in fact, require substantial effort to accumulate, even with assistance from computational tools (Bird, 2011). Therefore, the collection of raw data can be considered another time-sensitive task for which the tradeoffs with previously-discussed annotation efforts must contend.

It could be the case that more raw data for training could make up for additional annotation and FST development effort or make the LP procedure unnecessary. Figure 5 shows that that increased raw data does provide increasing gains, but they diminish after 200k tokens. The best performance is achieved by using more annotation and LP. Most importantly, however, removing either annotations or LP results in a significant decline in accuracy, such that even with 600k training tokens, we are unable to achieve the results of high annotation and LP using only 100k tokens.

### 5.5 Correcting existing annotations

For all of the ENG experiments, we also ran “oracle” experiments using gold tags for the same sentences or a tag dictionary containing the same number of type/tag entries as the annotator produced, but containing only the most frequent entries as determined by the gold-labeled corpus. Using this simulated “perfect annotator” data shows we lose accuracy due to annotator mistakes: for our experienced annotator and maximal FST, using 4 hours of types the oracle accuracy is 90.5 vs. 88.5 while using only tokens we see 83.9 vs.

81.5. This indicates that there are gains to be made by correcting mistakes in the annotations. This is true even after the point of diminishing returns on the learning curve, meaning that even when adding *more* annotations no longer improves performance, progress can still be made by correcting errors, so it may be reasonable to ask annotators to attempt to correct errors in their past annotations. Automated techniques for facilitating error identification can be employed for this (Dickinson and Meurers, 2003).

## 6 Conclusions and Future Work

Care must be taken when drawing conclusions from small-scale annotation studies such as those presented in this paper. Nonetheless, we have explored realistic annotation scenarios for POS-tagging for low-resource languages and found several consistent patterns. Most importantly, it is clear that type annotations are the most useful input one can obtain from a linguist—provided a semi-supervised algorithm for projecting that information reliably onto raw tokens is available. In a sense, this result validates the research trajectory of efforts of the past two decades put into learning taggers from tag dictionaries: papers have successively removed layers of unrealistic assumptions, and in doing so have produced pipelines for type-supervision that easily beat token-supervision prepared in comparable amounts of time.

The result of most immediate practical value is that we show it is possible to train effective POS-taggers on actual low-resource languages given only a relatively small amount of unlabeled text and a few hours of annotation by a non-native linguist. Instead of having annotators label full sentences as one might expect the natural choice would be, it is much more effective to simply extract a list of the most frequent word types in the language and concentrate efforts on annotating these types with their potential parts of speech. Furthermore, for languages with rich morphology, a morphological transducer can yield significant performance gains when large amounts of other annotated resources are unavailable. (And it never hurts performance.)

Finally, additional raw text does improve performance. However, using substantial amounts of raw text is unlikely to produce gains larger than only a few hours spent annotating types. Thus, when deciding whether to spend time locating

larger volumes of digitized text or to spend time annotating types, choose types.

Despite the consistent superiority of type annotations in our experiments, it of course may be the case that techniques such as active learning may better select sentences for token annotation, so this should be explored in future work.

## Acknowledgements

We thank Kyle Jerro, Vijay John, Jim Evans, Yoav Goldberg, Slav Petrov, and the reviewers for their assistance and feedback. This work was supported by the U.S. Department of Defense through the U.S. Army Research Office (grant number W911NF-10-1-0533) and through a National Defense Science and Engineering Graduate Fellowship for the first author. Experiments were run on the UTCS Mastodon Cluster, provided by NSF grant EIA-0303609.

## References

- Steven Abney and Steven Bird. 2010. The human language project: Building a universal corpus of the worlds languages. In *Proceedings of ACL*.
- Jason Baldridge and Alexis Palmer. 2009. How well does active learning actually work? Time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of EMNLP*, Singapore.
- Steven Bird. 2011. Bootstrapping the language archive: New prospects for natural language processing in preserving linguistic heritage. *Linguistic Issues in Language Technology*, 6.
- Silviu Cucerzan and David Yarowsky. 2002. Bootstrapping a multilingual part-of-speech tagger in one person-day. In *Proceedings of CoNLL*, Taipei, Taiwan.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of ACL-HLT*, Portland, Oregon, USA.
- Markus Dickinson and W. Detmar Meurers. 2003. Detecting errors in part-of-speech annotation. In *Proceedings of EACL*.
- Weiwei Ding. 2011. Weakly supervised part-of-speech tagging for Chinese using label propagation. Master’s thesis, University of Texas at Austin.
- Dan Garrette and Jason Baldridge. 2012. Type-supervised hidden Markov models for part-of-speech tagging with incomplete tag dictionaries. In *Proceedings of EMNLP*, Jeju, Korea.

- Dan Garrette and Jason Baldridge. 2013. Learning a part-of-speech tagger from two hours of annotation. In *Proceedings of NAACL*, Atlanta, Georgia.
- Yoav Goldberg, Meni Adler, and Michael Elhadad. 2008. EM can find pretty good HMM POS-taggers (when given a good start). In *Proceedings ACL*.
- Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings NAACL*.
- Kazi Saidul Hasan and Vincent Ng. 2009. Weakly supervised part-of-speech tagging for morphologically-rich, resource-scarce languages. In *Proceedings of EACL*, Athens, Greece.
- Lauri Karttunen. 2001. Applications of finite-state transducers in natural language processing. *Lecture Notes in Computer Science*, 2088.
- Julian Kupiec. 1992. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech & Language*, 6(3).
- Shen Li, João Graça, and Ben Taskar. 2012. Wiki-ly supervised part-of-speech tagging. In *Proceedings of EMNLP*, Jeju Island, Korea.
- Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of CICLing*.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2).
- Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2).
- Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proceedings ACL*.
- Sujith Ravi and Kevin Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *Proceedings of ACL-AFNLP*.
- Sujith Ravi, Ashish Vaswani, Kevin Knight, and David Chiang. 2010. Fast, greedy model minimization for unsupervised tagging. In *Proceedings of COLING*.
- Emmanuel Roche and Yves Schabes. 1995. Deterministic part-of-speech tagging with finite-state transducers. *Computational Linguistics*, 21(2).
- Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings EMNLP*, Cambridge, MA.
- Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. In *Transactions of the ACL*. Association for Computational Linguistics.
- Partha Pratim Talukdar and Koby Crammer. 2009. New regularized algorithms for transductive learning. In *Proceedings of ECML-PKDD*, Bled, Slovenia.
- Kristina Toutanova and Mark Johnson. 2008. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *Proceedings of NIPS*.

# Using subcategorization knowledge to improve case prediction for translation to German

Marion Weller<sup>1</sup> Alexander Fraser<sup>2</sup> Sabine Schulte im Walde<sup>1</sup>

<sup>1</sup>Institut für Maschinelle  
Sprachverarbeitung  
Universität Stuttgart

{wellermn|schulte}@ims.uni-stuttgart.de

<sup>2</sup>Centrum für Informations-  
und Sprachverarbeitung  
Ludwig-Maximilians-Universität München

fraser@cis.uni-muenchen.de

## Abstract

This paper demonstrates the need and impact of subcategorization information for SMT. We combine (i) features on source-side syntactic subcategorization and (ii) an external knowledge base with quantitative, dependency-based information about target-side subcategorization frames. A manual evaluation of an English-to-German translation task shows that the subcategorization information has a positive impact on translation quality through better prediction of case.

## 1 Introduction

When translating from a morphologically poor language to a morphologically rich language we are faced with two major problems: (i) the richness of the target-language morphology causes data sparsity problems, and (ii) information about morphological features on the target side is not sufficiently contained in the source language morphology.

We address these two problems using a two-step procedure. We first replace inflected forms by their stems or lemmas: building a translation system on a stemmed representation of the target side leads to a simpler translation task, and the morphological information contained in the source and target language parts of the translation model is more balanced. In the second step, the stemmed output of the translation is then inflected: the morphological features are predicted, and the inflected forms are generated using the stem and predicted morphological features.

In this paper, we focus on improving *case* prediction for noun phrases (NPs) in German translations. The NP feature *case* is extremely difficult to predict in German: while the NP features *gender* and *number* are part of the stem or

can be derived from the source-side input, respectively, the prediction of *case* requires information about the subcategorization of the entire clause. This is due to German being a less configurational language than English, which encodes grammatical relations (e.g. subject-hood, object-hood, etc.) through the position of constituents. German sentences exhibit a freer constituent order, and thus case is an important indicator of the grammatical functions of noun phrases. Correct case prediction is a crucial factor for the adequacy of SMT output, cf. the example in table 1 providing an erroneously inflected output (this is taken from a baseline “simple inflection prediction” system, cf. section 5.2). The translation of the English input sentence in terms of stems is perfectly acceptable; after the inflection step, however, the translation of NP<sub>4</sub> *ongoing military actions* represents a genitive modifier of the subject NP<sub>2</sub>, instead of a direct object NP of the verb *anordnen* (*to order*). The meaning is thus *why the government of the ongoing military actions ordered*, which has only one NP and is completely wrong.

The translation in table 1 needs verb subcategorization information. This is demonstrated by the invented examples (1) and (2):

- (1) [Der Mitarbeiter]<sub>NP<sub>nom</sub></sub> hat [den Bericht]<sub>NP<sub>acc</sub></sub> [dem Kollegen]<sub>NP<sub>dat</sub></sub> gegeben.  
[The employee]<sub>NP<sub>nom</sub></sub> gave [his colleague]<sub>NP<sub>dat</sub></sub> [the report]<sub>NP<sub>acc</sub></sub>
- (2) [Der Mitarbeiter]<sub>NP<sub>nom</sub></sub> hat [dem Bericht]<sub>NP<sub>dat</sub></sub> [des Kollegen]<sub>NP<sub>gen</sub></sub> zugestimmt.  
[The employee]<sub>NP<sub>nom</sub></sub> agreed [on the report]<sub>PP</sub> [of his colleague]<sub>PP</sub>

Both inflected sentences rely on the stem sequence [d Mitarbeiter] [d Bericht] [d Kollege] ⟨verb⟩, so the case assignment can only be determined by the verb: While *geben* (*to give*) has a strong preference for selecting a ditransitive subcategorization frame<sup>1</sup>, including an agentive subject (nomi-

<sup>1</sup>A ditransitive verb takes a subject and two objects.

<b>input</b>		[why] <sub>1</sub> [the government] <sub>2</sub> [ordered] <sub>3</sub> [the ongoing military actions] <sub>4</sub>
<b>output</b>	stemmed	[warum] <sub>1</sub> [d Regierung] <sub>2</sub> [d anhaltend militärisch Aktion] <sub>4</sub> [angeordnet] <sub>3</sub>
	inflected	[warum] <sub>1</sub> [ <b>die</b> Regierung] <sub>2</sub> [ <b>der</b> anhaltenden militärischen Aktionen] <sub>4</sub> [angeordnet] <sub>3</sub>

Table 1: Example for case confusion in SMT output when using a simple prediction system.

native case), a benefactive (dative case) and a patient (accusative case), *zustimmen* (to agree) has a strong preference for only selecting an agentive subject (nominative case) and an indirect object theme (dative case). So in the latter case the NP [*d Kollege*] cannot receive case from the verb and is instead the genitive modifier of the dative NP.

While for examples (1) and (2) knowledge about the syntactic verb subcategorization functions is sufficient to correctly predict the NP cases, examples (3) to (6) require subcategorization information at the syntax-semantic interface.

- (3) [Der Mitarbeiter]<sub>NPnom</sub> hat [dem Kollegen]<sub>NPdat</sub> [den Bericht]<sub>NPacc</sub> gegeben.
- (4) [Der Mitarbeiter]<sub>NPnom</sub> hat [den Bericht]<sub>NPacc</sub> [dem Kollegen]<sub>NPdat</sub> gegeben.
- (5) [Dem Kollegen]<sub>NPdat</sub> hat [der Mitarbeiter]<sub>NPnom</sub> [den Bericht]<sub>NPacc</sub> gegeben.
- (6) [Den Bericht]<sub>NPacc</sub> hat [der Mitarbeiter]<sub>NPnom</sub> [dem Kollegen]<sub>NPdat</sub> gegeben.

In all four examples, the verb and the participating noun phrases *Mitarbeiter* (employee), *Kollege* (colleague) and *Bericht* (report) are identical, and the noun phrases are assigned the same case. However, given that the stemmed output of the translation does not tell us anything about case features, in order to predict the appropriate cases of the three noun phrases, we either rely on ordering heuristics (such that the nominative NP is more likely to be in the beginning of the sentence (the German *Vorfeld*) than the accusative or dative NP, even though all three of these would be grammatical), or we need fine-grained subcategorization information beyond pure syntax. For example, both *Mitarbeiter* and *Kollege* would satisfy the agentive subject role of the verb *geben* better than *Bericht*, and *Bericht* is more likely to be the patient of *geben*.

The contribution of this paper is to improve the prediction of case in our SMT system by implementing and combining two alternative routes to integrate subcategorization information from the syntax-semantic interface: (i) We regard the translation as a function of the source language input, and project the syntactic functions of the English nouns to their German translations in the

SMT output. This subcategorization model is necessary when there are several plausible solutions for the syntactic functions of a noun in combination with a verb. For example, both *Mitarbeiter* and *Kollege* are plausible subjects and direct objects of the verb *geben*, so the information about these nouns' roles in the input sentence allows for disambiguation. (ii) The case of an NP is derived from an external knowledge base comprising quantitative, dependency-based information about German verb subcategorization frames and noun modification. The verb subcategorization information is not restricted to syntactic noun functions but models association strength for verb–noun pairs with regard to the entire subcategorization frame plus the syntactic functions of the nouns. For example, the database can tell us that while the verb *geben* is very likely to subcategorize a ditransitive frame, the verb *zustimmen* is very likely to subcategorize only a direct object, next to the obligatory subject (**subcat frame prediction**). Furthermore, we can retrieve the information that the noun *Bericht* is less likely to appear as subject of *geben* than the nouns *Mitarbeiter* and *Kollege* (**verb–noun subcat case prediction**). And we can look up that the noun *Aktion* is very unlikely to be a genitive modification of *Regierung* (cf. table 1), while *Kollege* is a plausible genitive modification of *Bericht* (**noun–noun modification case prediction**, cf. example (2)).

In summary, model (i) applies when there are no obvious preferences concerning verb–noun subcategorization or noun–noun modification. Model (ii) predicts case relying on the subcategorization and modification preferences. The combination of our two models approaches a simplified level of semantic role definition but only relies on dependency information that is considerably easier and cheaper to define and obtain than a very high quality semantic parser and/or a corpus annotated with semantic role information. Integrating semantic role information into SMT has been demonstrated by various researchers to improve translation quality (cf. Wu and Fung (2009a), Wu and Fung (2009b), Liu and Gildea (2008), Liu and Gildea (2010)). Our approach is in line with

Wu and Fung (2009b) who demonstrated that on the one hand 84% of verb syntactic functions in a 50-sentence test corpus projected from Chinese to English, and that on the other hand about 15% of the subjects were not translated into subjects, but their semantic roles were preserved across language. These two findings correspond to the expected uses of our models (i) and (ii), respectively.

## 2 Previous work

Previous work has already introduced the idea of generating inflected forms as a post-processing step for a translation system that has been stripped of (most) target-language-specific features. Toutanova et al. (2008) and Jeong et al. (2010) built translation systems that predict inflected word forms based on a large array of morphological and syntactic features, obtained from both source and target side. Kholy and Habash (2012) and Green and DeNero (2012) work on English to Arabic translation and model gender, number and definiteness, focusing primarily on improving fluency.

Fraser et al. (2012) used a phrase-based system to transfer stems and generated inflected forms based on the stems and their morphological features. For case prediction, they trained a CRF with access to lemmas and POS-tags within a given window. We re-implemented the system by Fraser et al. as a hierarchical machine translation system using a string-to-tree setup. In contrast to the flat phrase-based setting of Fraser et al. (2012), syntactic trees on the SMT output allow us to work with verb–noun structures, which are relevant for case prediction. While the CRF used for case prediction in Fraser et al. (2012) has access to lexical information, it is limited to a certain window size and has no direct information about the relation of verb–noun pairs occurring in the sentence. Using a window of a limited size is particularly problematic for German, as there can be large gaps between the verb and its subcategorized nouns; introducing information about the relation of verbs and nouns helps to bridge such gaps. Furthermore, that model was not able to make effective use of source-side features.

One of the objectives of using an inflection prediction model is morphologically well-formed output. Kirchhoff et al. (2012) evaluated user reactions to different error types in machine translation and came to the result that morphological

well-formedness has only a marginal impact on the comprehensibility of SMT output in the case of English-Spanish translation. As already discussed, German *case* is essential to the meaning of the sentence, so this result will not hold for German output.

## 3 Translation pipeline

This section presents an overview of our two-step translation process. In the first step, English input is translated to German stems. In the second step, morphological features are predicted and inflected forms are generated based on the word stems and the morphological features. In subsections 3.1 to 3.4, we present the simple version of the inflection prediction system; our new features are described in sections 4.2 and 4.3.

### 3.1 Stemmed representation/feature markup

We first parse the German side of the parallel training data with BitPar (Schmid, 2004). This maps each surface form appearing in normal text to a stem and morphological features (case, gender, number). We use this representation to create the stemmed representation for training the translation model. With the exception of stem-markup (discussed below), all morphological features are removed from the stemmed representation. The stem markup is used as part of the input to the feature prediction; the basic idea is that the given feature values are picked up by the prediction model and then propagated over the phrase.

Nouns, as the head of NPs and PPs, are annotated with *gender* and *number*. We consider gender as part of the stem, whereas the value for number is derived from the source-side: if marked for number, singular/plural nouns are distinguished during word alignment and then translated accordingly. Prepositions are also annotated with case; many prepositions are restricted to only one case, some are ambiguous and allow for either *dative* or *accusative*. Other words which are subject to feature prediction (e.g. adjectives, articles) are reduced to their stems with no feature markup, as are all remaining words. As sole exception, we keep the inflected forms of verbs (verbal inflection is not modelled). In addition to the translation model, the target-side language model, as well as the reference data for parameter tuning use this representation.

### 3.2 Building a stemmed translation model

We use a hierarchical translation system. Instead of translating phrases, a hierarchical system extracts translation rules (Galley et al., 2004) which allow the decoder to provide a tree spanning over the translated sentence. In order to avoid sparsity during rule extraction, we use a string-to-tree setup, where only the target-side part of the data is parsed. Translation rules are of the following form:

```
[X]1 allows [X]2 → [NP]1 [NP]2 erlaubt  
[X]1 allows [X]2 → [NP]1 erlaubt [NP]2
```

This example illustrates how rules can cover the different word ordering possibilities in German.

PP nodes are annotated with their respective case, as well as with the lemma of the preposition they contain. In our experiments, this enriched annotation has small improvements over the simpler setting with only head categories (details omitted). This outcome, in particular that adding the lemma of the preposition to the PP node helps to improve translation quality, has been observed before in tree restructuring work for improving translation (Huang and Knight, 2006).

### 3.3 Feature prediction and generation of inflected forms

In this section we discuss our focus, which is prediction of case, but also the prediction of number, gender and strong/weak adjectival inflection. The latter feature is German-specific; its values<sup>2</sup> (strong/weak) depend on the combination of the other features, as well as on the type of determiner (e.g. definite/indefinite/none).

Morphological features are predicted on four separate CRF models, one for each feature. The models for case, number and gender are independent of another, whereas the model for adjectival inflection requires information about these features, and is thus the last one to be computed, taking the output of the 3 other models as part of its input. In contrast, the adjectival inflection model in Fraser et al. (2012) is independent from the other features. Each model has access to stems, POS-tags and the feature to be modelled within a window of four positions to the right and the left of the current position<sup>3</sup>.

<sup>2</sup>Note that the values for strong/weak inflection are not always the same over the phrase, but follow a certain pattern depending on the settings of case, number and gender.

<sup>3</sup>Preliminary experiments showed that larger windows do not improve translation quality.

Table 2 illustrates the different steps of the inflection process: the markup (number and gender on nouns) in the stemmed output of the SMT system is part of the input to the respective feature prediction. For gender and number, the values given on the stems of the nouns are then propagated over the phrase. While the case of prepositional phrases is determined by the case annotation on prepositions, the case of nominal phrases is computed only based on the respective contexts. After predicting all morphological features, the information required to generate inflected forms is complete: based on the stems and the features, we use the morphological tool SMOR (Schmid et al., 2004) for the generation of inflected forms.

One general problem with feature-prediction is that the ill-formed SMT output is not well represented by the training data which consists of well-formed sentences. This problem was also mentioned by Stymne and Cancedda (2011) and Kholy and Habash (2012). They deal with this problem by translating the training data and annotating it with the respective features, and then adding this new data set to the original training data. As this method comes with its own problems, such as transferring the morphological annotation to not necessarily isomorphically translated text, we do not use translated data as part of the training data. Instead, we limit the power of the CRF model through experimenting with the removal of features, until we had a system that was robust to this problem.

### 3.4 Dealing with word formation issues

To reduce data sparsity, we split portmanteau prepositions. Portmanteaus are compounds of prepositions and articles, e.g. *zur* = *zu der* (*to the*). Being components of nominal phrases, they have to agree in all morphological features with the rest of the phrase. As only some combinations of articles and prepositions can form a portmanteau, the decision of whether to merge prepositions and articles is made after feature prediction. Since our focus is case prediction, we do not do special modelling of German compounds.

## 4 Using subcategorization information

Within the area of (automatic) lexical acquisition, the definition of lexical verb information has been a major focus, because verbs play a central role for the structure and the meaning of sentences and

SMT output	predicted features	inflected forms	gloss
beeinflussen<VFIN>	–	beeinflussen	influence
d<ART>	Fem.Acc.Sg.St	die	the
politisch<ADJ>	Fem.Acc.Sg.Wk	politische	political
Stabilität<NN><Fem><Sg>	Fem.Acc.Sg.Wk	Stabilität	stability

Table 2: Overview of the inflection process: the stem markup is highlighted in the SMT output.

discourse. On the one hand, this has led to a range of manually or semi-automatically developed lexical resources focusing on verb information, such as the Levin classes (Levin, 1993), VerbNet (Kipper Schuler, 2006), FrameNet<sup>4</sup> (Fillmore et al., 2003), and PropBank (Palmer et al., 2005). On the other hand, we find automatic approaches to the induction of verb subcategorization information at the syntax-semantics interface for a large number of languages, e.g. Briscoe and Carroll (1997) for English; Sarkar and Zeman (2000) for Czech; Schulte im Walde (2002a) for German; Messiant (2008) for French. This basic kind of verb knowledge has been shown to be useful in many NLP tasks such as information extraction (Surdeanu et al., 2003; Venturi1 et al., 2009), parsing (Carroll et al., 1998; Carroll and Fang, 2004) and word sense disambiguation (Kohomban and Lee, 2005; McCarthy et al., 2007).

#### 4.1 Extracting subcategorization information

As described in the introductory section, we make use of two<sup>5</sup> major kinds of subcategorization information. Verb–noun tuples referring to specific syntactic functions within verb subcategorization (**verb–noun subcat case prediction**) are integrated with an associated probability for *accusative* (direct object), *dative* (indirect object) and *nominative* (subject).<sup>6</sup> Further to the subject and object noun phrases, the subcategorization information provides quantitative triples for verb–preposition–noun pairs, thus predicting the case of NPs within prepositional phrases (we do this only when the prepositions are ambiguous, i.e., they could subcategorize either a dative or an accusative NP). In addition to modelling subcategorization information, it is also important to differentiate between subcategorized noun phrases (such as object or subject), and noun phrases

<sup>4</sup>Even though the FrameNets approach does not only include knowledge about verbal predicates, the actual lexicons are skewed towards verb behaviour.

<sup>5</sup>The third kind of information, **subcat frame prediction** is implicit, since verb–noun tuples rely on specific frames.

<sup>6</sup>Genitive objects can also occur in German verb subcategorization frames, but this is extremely rare and verb-specific and thus not considered in our model.

	V-SUBJ	V-OBJ <sub>Acc</sub>	V-OBJ <sub>Dat</sub>
EP	454,350	332,847	53,711
HGC	712,717	329,830	160,377
Both	1,089,492	607,541	206,764

Table 3: Number of verb-noun types extracted from Europarl (EP) and newspaper data (HGC).

that modify nouns (**noun–noun modification case prediction**). Typically, these NP modifiers are genitive NPs. To this end, we integrate noun–noun<sub>Gen</sub> tuples with their respective frequencies. These preferences for a certain function (i.e. subject, object or modifier) are passed on to the system at the level of nouns and integrated into the CRF through the derived probabilities.

The tuples and triples are obtained from dependency-parsed data by extracting all occurrences of the respective relations; table 3 gives an overview of the number of extracted tuple types. For the subcategorization information, the verb–noun tuples (verb–subject, verb–object<sub>Acc</sub>, verb–object<sub>Dat</sub>) are then grouped as follows:

tuple	gloss	Acc	Dat	Nom
Schema <sub>N</sub> folgen <sub>V</sub>	pattern follow	0	322	19

We compute the probabilities for the verb–noun tuple to occur in the respective functions based on the relative frequencies. In the case of *Schema<sub>N</sub> folgen<sub>V</sub>*, we find that the function of *Schema* as dative object is predominant (*to follow a pattern*), but it can also occur in the subject position (*the pattern follows*). The fact that two functions are possible for this noun are reflected in their probabilities. The probabilities are discretized into 5 buckets ( $B_{p=0}$ ,  $B_{0<p\leq 0.25}$ ,  $B_{0.25<p\leq 0.5}$ ,  $B_{0.5<p\leq 0.75}$ ,  $B_{0.75<p\leq 1}$ ). In contrast, noun modification in noun–noun<sub>Gen</sub> construction is represented by co-occurrence frequencies.<sup>7</sup>

<sup>7</sup>The frequencies are bucketed to the powers of ten, i.e.  $f = 1, 2 \leq f \leq 10, 11 \leq f \leq 100$ , etc. and also  $f = 0$ : this representation allows for a more fine-grained distinction in the low-to-mid frequency range, providing a good basis for the decision of whether a given noun–noun pair is a true noun–noun<sub>Gen</sub> structure or just a random co-occurrence of two nouns.

	Gloss	Stem	Tag	Acc	Dat	Nom	Verb	Gen	NI	Gold
1	<i>companies</i>	Unternehmen<NN>	NN	0.00	0.00	1.00	erhalten	-	-	Nom
2	<i>should</i>	sollten<VVFIN>	VVFIN	-	-	-	-	-	-	-
3	<i>financial</i>	finanziell<ADJ>	ADJ	-	-	-	-	-	-	Acc
4	<i>funding</i>	Mittel<NN>	NN	1.00	0.00	0.00	erhalten	-	-	Acc
5	<i>for</i>	für APPR<Acc>	PRP	-	-	-	-	-	-	-
6	<i>the</i>	d<ART>	ART	-	-	-	-	-	-	Acc
7	<i>introduction</i>	Einführung<NN>	NN	-	-	-	-	-	-	Acc
8	<i>new</i>	neu<ADJ>	ADJ	-	-	-	-	-	-	Gen
9	<i>technologies</i>	Technologie<NN>	NN	-	-	-	-	100	Einführung<NN>	Gen
10	<i>obtain</i>	erhalten<VVFIN>	VVFIN	-	-	-	-	-	-	-

Table 4: Adding subcategorization information into SMT output. (EN input: *companies should obtain financial funding for the introduction of new technologies*). On the right, the correct labels are given.

## 4.2 Integrating subcategorization knowledge

There are two possibilities to integrate subcategorization information into the case prediction model: (i) It can be integrated into the data set using the tree-structure provided by the decoder. Here, verb-noun tuples are extracted from VP and S structures, and then the probabilities for the different functions are looked up. Similarly, for two adjacent NPs, the occurrence frequencies of the respective two nouns are looked up in the list of noun-noun<sub>Gen</sub> constructions. (ii) The subcategorization information can be integrated based on the verb-noun tuples obtained by using tuples obtained from source-side dependencies.

The classification task of the CRF consists in predicting a sequence of labels: case values for NPs/PPs or no value otherwise, cf. table 4. The model has access to the basic features *stem* and *tag*, as well as the new features based on subcategorization information (explained below), using unigrams within a window of up to four positions to the right and the left of the current position, as well as bigrams and trigrams for stems and tags (current item + left and/or right item).

An example for integrating subcategorization features is given in table 4. The first word *Unternehmen* (*companies*) is annotated as subject of *erhalten* (*obtain*) with probability 1, and *Mittel* (*funding*) is annotated as direct object of *erhalten* with probability 1. The word *Technologie* (*technology*) has been marked as a candidate for a genitive in a noun-noun<sub>Gen</sub> construction<sup>8</sup>; the co-occurrence frequency of the tuple *Einführung-Technologie* (*introduction - technology*) lies in the bucket 11...100.

In addition to the probability/frequency of the respective functions, we also provide the CRF with bigrams containing the two parts of the tuple,

<sup>8</sup>There is no annotation on *Einführung* as the preposition *für* is always in accusative case.

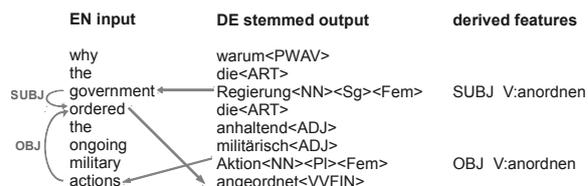


Figure 1: Deriving features from dependency-parsed English data via the word alignment.

i.e. verb+noun or the two nouns of possible noun-noun<sub>Gen</sub> constructions. As can be seen in the example in table 4, the subject (line 1) and the verb (line 10) are far apart from each other. By providing the parts of the tuple as unigrams, bigrams or trigrams to the CRF, all relevant information is available: verb, noun and the probabilities for the potential functions of the noun in the sentence. In addition to bridging the long distance between verbs and subcategorized nouns, a very common problem for German, this type of precise information also helps to close the gap between the well-formed training data and the broken SMT-output as it replaces to a certain extent the target-language context information (n-grams of stems or lemmas within a small window).

## 4.3 Integrating source-side features

For predicting case in SMT output, information about an NP's function in the input sentence is essential. Syntax-semantic functions can be isomorphic (e.g., English subjects and objects may have the same function in a German translation), but this is not necessarily the case. Despite this, an important advantage of integrating source-side features is that the well-formed source-side text can be reliably parsed, whereas SMT output is often disfluent and cannot be reliably parsed.

The English features are obtained from dependency-parsed data (Choi and Palmer, 2012). The relevant annotation of the parser is transferred

to the SMT output via word alignment. We focus on English subjects, direct objects and noun-of-noun structures (often equivalent to noun-noun<sub>Gen</sub> phrases on the German side): these structures are generally likely to correspond to each other within source and target language. In contrast to the subcategorization-based information, the difference between well-formed training data and disfluent SMT output tends to work to our benefit here: while the parallel sentences of the training data were manually translated with the objective to produce good target-language sentences, the syntactic structures of the source and target sentences are often diverging. In contrast, the SMT system often produces more isomorphic translations, which is helpful for annotating source-side features on the target language.

Figure 1 shows the process of integrating source-side features: for each German noun that is aligned with an English noun labelled as subject or direct object, this annotation is transferred to the target-side. Using the English dependency structures, the verb subcategorizing the respective noun is identified, and via the alignment, the equivalent German verb is obtained. Similarly, candidates for noun-noun<sub>Gen</sub> structures are identified by extracting and aligning English noun-of-noun phrases.

## 5 Experiments and evaluation

In this section, we present experiments using different feature combinations. We also present a manual evaluation of our best system which shows that the new features improve translation quality.

### 5.1 Data and experimental setup

We use the hierarchical translation system that comes with the Moses SMT-package and GIZA++ to compute the word alignment, using the “grow-diag-final-and” heuristics. The rule table was computed with the default parameter setting for GHKM extraction (Galley et al., 2004) in the implementation by Williams and Koehn (2012).

Our training data contains 1,485,059 parallel sentences<sup>9</sup>; the German part of the parallel data is used as the target-side language model. The dev and test sets (1025/1026 lines) are wmt-2009-a/b.

For predicting the grammatical features, we used the Wapiti Toolkit (Lavergne et al., 2010).<sup>10</sup>

<sup>9</sup>English/German data released for the 2009 ACL Workshop on Machine Translation shared task.

<sup>10</sup>To eliminate irrelevant features, we use L1 regulariza-

We train four CRFs on data prepared as shown in section 3. The corpora used for the extraction of subcategorization tuples were Europarl and German newspaper data (200 million words). We choose this particular data combination in order to provide data that matches the training data, as well as to add new data of the test set’s domain (news). The German part of Europarl was dependency-parsed with Bohnet (2010), and subcategorization information was extracted as described in Scheible et al. (2013); the newspaper data (HGC - Huge German Corpus) was parsed with Schmid (2000), and subcategorization information was extracted as described in Schulte im Walde (2002b).

### 5.2 Results

We report results of two types of systems (table 5): first, a regular translation system built on surface forms (i.e., normal text) and second, four inflection prediction systems. The first inflection prediction system (1) uses a simple case prediction model, whereas the remaining systems are enriched with (2) subcategorization information (cf. section 4.2), (3) source-side features (cf. section 4.3), and (4) both source-side features and subcategorization information. In (2) and (4), the subcategorization information was included using tuples obtained from source-side dependencies<sup>11</sup>. The simple prediction system corresponds to that presented in section 3; for all inflection prediction systems, the same SMT output and models for number, gender and strong/weak inflection were used; thus the only difference with the simple prediction system is the model for case prediction.

We present three types of evaluation: BLEU scores (Papineni et al., 2001), prediction accuracy on clean data and a manual evaluation of the best system in section 5.3.

Table 5 gives results in case-insensitive BLEU. While the inflection prediction systems (1-4) are significantly<sup>12</sup> better than the surface-form system (0), the different versions of the inflection systems are not distinguishable in terms of BLEU; however, our manual evaluation shows that the new features have a positive impact on translation quality.

tion; the regularization parameter is optimized on held out data.

<sup>11</sup>Using tuples extracted from the target-side parse tree (produced by the decoder) results in a BLEU score of 14.00.

<sup>12</sup>We used Kevin Gimpel’s implementation of pairwise bootstrap resampling with 1000 samples.

	0	1	2	3	4
	<b>surface system</b>	<b>simple prediction</b>	<b>subcat. features (tuples from EN side)</b>	<b>source-side features</b>	<b>source-side + subcat. features</b>
<b>BLEU</b>	13.43	14.02	14.05	14.10	14.17
<b>Clean</b>	–	85.05 %	85.65 %	85.61 %	85.81 %

Table 5: Results of the simple prediction vs. three systems enriched with extra features.

One problem with using BLEU as an evaluation metric is that it is a precision-oriented metric and tends to reward fluency rather than adequacy (see (Wu and Fung, 2009a; Liu and Gildea, 2010)). As we are working on improving adequacy, this will not be fully reflected by BLEU. Furthermore, not all components of an NP do necessarily change their inflection with a new case value; it might happen that the only indicator for the case of an NP is the determiner: *er sieht [den alten Mann]<sub>NPacc</sub>* (he sees the old man) vs. *er folgt [dem alten Mann]<sub>NPdat</sub>* (he follows the old man). While the case marking of NPs is essential for comprehensibility, one changed word per noun phrase is hardly enough to be reflected by BLEU.

An alternative to study the effectiveness of the case prediction model is to evaluate the prediction accuracy on parsed clean data, i.e. not on SMT output. In this case, we measure (using the dev set) how often the case of an NP is predicted correctly<sup>13</sup>. In all cases, the prediction accuracy is better for the enriched systems. This shows that the additional features improve the model, but also that a gain in prediction accuracy on clean data is not necessarily related to a gain in BLEU. We observed that the more complex the model, the less robust it is to differences between the test data and the training data. Related to this problem, we observed that high-order n-gram POS/lemma-based features in the simple prediction (sequences of lemmas and tags) are given too much weight in training and thus make it difficult for the new features to have a larger impact, so we restricted the n-gram order of this type of feature to trigrams.

### 5.3 Manual evaluation of the best system

In order to provide a better understanding of the impact of the presented features, in particular to see whether there is an improvement in adequacy, we carried out a manual evaluation comparing sys-

<sup>13</sup>The numbers in table 5 are artificially high and downplay the difference as they also include cases which are very easy to predict, such as nouns in PPs where only one value for case is possible. We measure how many case labels were correctly predicted, not correct inflected forms.

		<b>enriched preferred</b>	<b>simple preferred</b>	<b>equal</b>
(a)	<b>person 1</b>	23	11	12
	<b>person 2</b>	21	8	17
	<b>person 3</b>	26	11	9
(b)	<b>person 1</b>	23	5	18
	<b>person 2</b>	21	11	14
	<b>person 3</b>	29	8	9
(c)	<b>agreement</b>	17	2	6

Table 6: Manual evaluation of 46 sentences: without (a) and with (b) access to EN input, and the annotators’ agreement in the second part (c).

tem (4) with the simple prediction system (1). From the set of different sentences between the simple prediction system and the enriched system (144 of 1026), we evaluated those where the English input sentence was between 8 and 25 words long (46 sentences in total). We specifically restricted the test set in order to provide sentences which are less difficult to annotate, as longer sentences are often very disfluent and too hard to rate. Most of the sentences in the evaluation set differ only in the realization of one NP. For comparing the two systems, the sentences were presented in random order to 3 native speakers of German.

The evaluation consists of two parts: first, the participants were asked to decide which sentence is better without being given the English input (this measures fluency). In the second part, they should to mark that sentence which better reproduces the content of the English input sentence (this measures adequacy). The test set is the same for both tasks, the only difference being that the English input is given in the second part. The results are given in table 6. Summarizing we can say that the participants prefer the enriched system over the simple system in both parts; there is a high agreement (17 cases) in decisions over those sentences which were rated as *enriched better*.

When looking at the pairwise inter-annotator agreement for the task of annotating the test-set with the 3 possible labels *enriched preferred*, *simple preferred* and *no preference*, we find that the annotators P1 and P2 have a substantial agreement

1	<b>input simple enriched</b>	hundreds of policemen were on alert , and [a helicopter] <sub>Subj</sub> circled the area with searchlights . Hunderte von Polizisten auf Trab , und [einen Helikopter] <sub>Acc</sub> eingekreist das Gebiet mit searchlights . Hunderte von Polizisten auf Trab , und [ein Helikopter] <sub>Nom</sub> eingekreist das Gebiet mit searchlights .
2	<b>input simple enriched</b>	while 38 %percent put [their trust] <sub>Obj</sub> in viktor orbán . während 38 % [ihres Vertrauens] <sub>Gen</sub> schenken in Viktor Orbán . während 38 % [ihr Vertrauen] <sub>Acc</sub> schenken in Viktor Orbán .
3	<b>input simple enriched</b>	more than \$ 100 billion will enter [the monetary markets] <sub>Obj</sub> by means of public sales . mehr als 100 Milliarden Dollar werden durch öffentlichen Verkauf [der Geldmärkte] <sub>Gen</sub> treten . mehr als 100 Milliarden Dollar werden durch öffentlichen Verkauf [die Geldmärkte] <sub>Acc</sub> treten .

Table 7: Output from the simple system (1) and the enriched system (4).

in terms of Kappa ( $\kappa = 0.6184$ ), whereas the agreement of P3 with P1/P2 respectively leads to lower scores ( $\kappa = 0.4467$  and  $\kappa = 0.3596$ ). However, the annotators tend to agree well on sentences with the label *enriched preferred*, but largely disagree on sentences labelled as either *simple preferred* or *no preference*. The number of decisions where all three annotators agree on a label when given the English input is listed in table 6(c): for example, only two sentences were given the label *baseline is better* by all three annotators. This outcome shows how difficult it is to rate disfluent SMT output. For evaluating the case prediction system, the distinction between *enriched preferred* and *enriched dispreferred* is the most important question to answer. Redefining the annotation task to annotating only two values by grouping the labels *simple preferred* and *no preference* into one annotation possibility leads to  $\kappa = 0.7391$ ,  $\kappa = 0.4048$  and  $\kappa = 0.5652$ .

#### 5.4 Examples

Table 7 shows some examples for output from the simple system and the system using source-side and subcategorization features. In the first sentence, the subject NP *a helicopter* was inflected as a direct object in the simple system, but as a subject in the enriched system, which was preferred by all three annotators. In the second sentence, the NP *their trust*, i.e. a direct object of *put*, was incorrectly predicted as genitive-modifier of *38 %* (i.e. *38 % of their trust*) in the simple system. The enriched system made use of the preference for accusative for the pair *Vertrauen schenken* (*place trust*), correctly inflecting this NP as direct object. Interestingly, only two annotators preferred the enriched system, whereas one was undecided. The third sentence illustrates how difficult it is to rate case marking on disfluent SMT output: there are two possibilities to translate *enter the money market*; the direct equivalent of the English phrase (*den Geldmarkt<sub>Acc</sub> betreten*), or via the use

of a prepositional phrase (*auf den Geldmarkt<sub>Acc</sub> treten*: “*to step into the money market*”). The SMT-output contains a mix of both, i.e. the verb *treten* (instead of *betreten*), but without the preposition, which cannot lead to a fully correct inflection. While the inflection of the simple system (a genitive construction meaning *the public sales of the money market*) is definitely wrong, the inflection obtained in the enriched system is not useful either, due to the structure of the translation<sup>14</sup>. This difficulty is also reflected by the annotators, who gave twice the label *no preference* and once the label *enriched better*.

## 6 Conclusion

We illustrated the necessity of using external knowledge sources like subcategorization information for modelling case for English to German translation. We presented a translation system making use of a subcategorization database together with source-side features. Our method is language-independent with regard to the source language; furthermore, no language-specific high-quality semantic annotation is needed for the target language, but the data required to model the subcategorization preferences can be obtained using standard NLP techniques. We showed in a manual evaluation that the proposed features have a positive impact on translation quality.

## Acknowledgements

This work was funded by the DFG Research Project *Distributional Approaches to Semantic Relatedness* (Marion Weller), the DFG Heisenberg Fellowship SCHU-2580/1-1 (Sabine Schulte im Walde), as well as by the Deutsche Forschungsgemeinschaft grant *Models of Morphosyntax for Statistical Machine Translation* (Alexander Fraser).

<sup>14</sup>Furthermore, with *treten* being polysemous, *die Geldmärkte treten* can also mean *to kick the money markets*.

## References

- Bernd Bohnet. 2010. Top Accuracy and Fast Dependency Parsing is not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING) 2010*, pages 89–97, Beijing, August.
- Ted Briscoe and John Carroll. 1997. Automatic Extraction of Subcategorization from Corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, pages 356–363, Washington, DC.
- John Carroll and Alex C. Fang. 2004. The Automatic Acquisition of Verb Subcategorisations and their Impact on the Performance of an HPSG Parser. In *Proceedings of the 1st International Joint Conference on Natural Language Processing*, pages 107–114, Sanya City, China.
- John Carroll, Guido Minnen, and Ted Briscoe. 1998. Can Subcategorisation Probabilities Help a Statistical Parser? In *Proceedings of the 6th ACL/SIGDAT Workshop on Very Large Corpora*, Montreal, Canada.
- Jinho D. Choi and Martha Palmer. 2012. Getting the Most out of Transition-Based Dependency Parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Charles J. Fillmore, Christopher R. Johnson, and Miriam R.L. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16:235–250.
- Alexander Fraser, Marion Weller, Aoife Cahill, and Fabienne Cap. 2012. Modeling Inflection and Word-Formation in SMT. In *Proceedings of the the European Chapter of the Association for Computational Linguistics (EACL)*, Avignon, France.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a Translation Rule? In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT-NAACL)*.
- Spence Green and John DeNero. 2012. A Class-Based Agreement Model for Generating Accurately Inflected Translations. pages 146–155.
- Bryant Huang and Kevin Knight. 2006. Relabeling Syntax Trees to Improve Syntax-Based Machine Translation Quality. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.
- Minwoo Jeong, Kristina Toutanova, Hisami Suzuki, and Chris Quirk. 2010. A Discriminative Lexicon Model for Complex Morphology. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*.
- Ahmed El Kholy and Nizar Habash. 2012. Translate, Predict or Generate: Modeling Rich Morphology in Statistical Machine Translation. In *European Association for Machine Translation*.
- Karin Kipper Schuler. 2006. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania, Computer and Information Science.
- Katrin Kirchhoff, Daniel Capurro, and Anne Turner. 2012. Evaluating User Preferences in Machine Translation Using Conjoint Analysis. In *European Association for Machine Translation*.
- Upali S. Kohomban and Wee Sun Lee. 2005. Learning Semantic Classes for Word Sense Disambiguation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 34–41, Ann Arbor, MI.
- Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513. Association for Computational Linguistics, July.
- Beth Levin. 1993. *English Verb Classes and Alternations*. The University of Chicago Press.
- Ding Liu and Daniel Gildea. 2008. Improved Tree-to-String Transducers for Machine Translation. In *ACL Workshop on Statistical Machine Translation*.
- Ding Liu and Daniel Gildea. 2010. Semantic Role Features for Machine Translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING) 2010*.
- Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2007. Unsupervised Acquisition of Predominant Word Senses. *Computational Linguistics*, 33(4):553–590.
- Cédric Messiant. 2008. A Subcategorization Acquisition System for French Verbs. In *Proceedings of the Student Research Workshop at the 46th Annual Meeting of the Association for Computational Linguistics*, pages 55–60, Columbus, OH.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated Resource of Semantic Roles. *Computational Linguistics*, 31(1):71–106.
- Kishore A. Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: a Method for Automatic Evaluation of Machine Translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center.
- Anoop Sarkar and Daniel Zeman. 2000. Automatic Extraction of Subcategorization Frames for Czech. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrücken, Germany.

- Silke Scheible, Sabine Schulte im Walde, Marion Weller, and Max Kisselew. 2013. A Compact but Linguistically Detailed Database for German Verb Subcategorisation relying on Dependency Parses from a Web Corpus. In *Proceedings of the 8th Web as Corpus Workshop*, Lancaster, UK. To appear.
- Helmut Schmid, Arne Fitschen, and Ulrich Heid. 2004. SMOR: a German Computational Morphology Covering Derivation, Composition, and Inflection. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*.
- Helmut Schmid. 2000. LoPar: Design and Implementation. Arbeitspapiere des Sonderforschungsbereichs 340 'Linguistic Theory and the Foundations of Computational Linguistics' 149, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.
- Helmut Schmid. 2004. Efficient Parsing of Highly Ambiguous Context-Free Grammars with Bit Vectors.
- Sabine Schulte im Walde. 2002a. A Subcategorisation Lexicon for German Verbs induced from a Lexicalised PCFG. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*, volume IV, pages 1351–1357, Las Palmas de Gran Canaria, Spain.
- Sabine Schulte im Walde. 2002b. A Subcategorisation Lexicon for German Verbs induced from a Lexicalised PCFG. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*, volume IV, pages 1351–1357, Las Palmas de Gran Canaria, Spain.
- Sara Stymne and Nicola Cancedda. 2011. Productive Generation of Compound Words in Statistical Machine Translation. In *Proceedings of the Sixth Workshop on Machine Translation*.
- Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using Predicate-Argument Structures for Information Extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 8–15, Sapporo, Japan.
- Kristina Toutanova, Hisami Suzuki, and Achim Ruopp. 2008. Applying Morphology Generation Models to Machine Translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL): Human Language Technologies*.
- Giulia Venturi<sup>1</sup>, Simonetta Montemagni, Simone Marchi, Yutaka Sasaki, Paul Thompson, John McNaught, and Sophia Ananiadou. 2009. Bootstrapping a Verb Lexicon for Biomedical Information Extraction. In Alexander Gelbukh, editor, *Linguistics and Intelligent Text Processing*, pages 137–148. Springer, Heidelberg.
- Philip Williams and Phillipp Koehn. 2012. GHKM-Rule Extraction and Scope-3 Parsing in Moses. In *Proceedings of the 7th Workshop on Statistical Machine Translation, ACL*.
- Dekai Wu and Pascale Fung. 2009a. Can Semantic Role Labeling Improve SMT? In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT)*.
- Dekai Wu and Pascale Fung. 2009b. Semantic Roles for SMT: A Hybrid two-pass Model. In *Proceedings of the North American Chapter of the Association for Computational Linguistics and Human Language Technologies Conference (NAACL-HLT)*.

# Name-aware Machine Translation

Haibo Li<sup>†</sup> Jing Zheng<sup>‡</sup> Heng Ji<sup>†</sup> Qi Li<sup>†</sup> Wen Wang<sup>‡</sup>

<sup>†</sup> Computer Science Department and Linguistics Department  
Queens College and Graduate Center, City University of New York  
New York, NY, USA 10016  
{lihaibo.c, hengjicuny, liqiearth}@gmail.com

<sup>‡</sup> Speech Technology & Research Laboratory  
SRI International  
Menlo Park, CA, USA 94025  
{zj, wwang}@speech.sri.com

## Abstract

We propose a Name-aware Machine Translation (MT) approach which can tightly integrate name processing into MT model, by jointly annotating parallel corpora, extracting name-aware translation grammar and rules, adding name phrase table and name translation driven decoding. Additionally, we also propose a new MT metric to appropriately evaluate the translation quality of informative words, by assigning different weights to different words according to their importance values in a document. Experiments on Chinese-English translation demonstrated the effectiveness of our approach on enhancing the quality of overall translation, name translation and word alignment over a high-quality MT baseline<sup>1</sup>.

## 1 Introduction

A shrinking fraction of the world's Web pages are written in English, therefore the ability to access pages across a range of languages is becoming increasingly important. This need can be addressed in part by cross-lingual information access tasks such as entity linking (McNamee et al., 2011; Cassidy et al., 2012), event extraction (Hakkani-Tur et al., 2007), slot filling (Snover et al., 2011) and question answering (Parton et al., 2009; Parton and McKeown, 2010). A key bottleneck of high-quality cross-lingual information access lies in the performance of Machine Translation (MT). Traditional MT approaches focus on the fluency and accuracy of the overall translation but fall short in their ability to translate certain content words including critical information, especially names.

<sup>1</sup>Some of the resources and open source programs developed in this work are made freely available for research purpose at <http://nlp.cs.qc.cuny.edu/NAMT.tgz>

A typical statistical MT system can only translate 60% person names correctly (Ji et al., 2009). Incorrect segmentation and translation of names which often carry central meanings of a sentence can also yield incorrect translation of long contexts. Names have been largely neglected in the prior MT research due to the following reasons:

- The current dominant automatic MT scoring metrics (such as Bilingual Evaluation Understudy (BLEU) (Papineni et al., 2002)) treat all words equally, but names have relative low frequency in text (about 6% in newswire and only 3% in web documents) and thus are vastly outnumbered by function words and common nouns, etc..
- Name translations pose a greater complexity because the set of names is open and highly dynamic. It is also important to acknowledge that there are many fundamental differences between the translation of names and other tokens, depending on whether a name is rendered phonetically, semantically, or a mixture of both (Ji et al., 2009).
- The artificial settings of assigning low weights to information translation (compared to overall word translation) in some large-scale government evaluations have discouraged MT developers to spend time and explore resources to tackle this problem.

We propose a novel Name-aware MT (NAMT) approach which can tightly integrate name processing into the training and decoding processes of an end-to-end MT pipeline, and a new name-aware metric to evaluate MT which can assign different weights to different tokens according to their importance values in a document. Compared to previous methods, the novel contributions of our approach are:

1. Tightly integrate joint bilingual name tagging into MT training by coordinating tagged

names in parallel corpora, updating word segmentation, word alignment and grammar extraction (Section 3.1).

2. Tightly integrate name tagging and translation into MT decoding via name-aware grammar (Section 3.2).
3. Optimize name translation and context translation simultaneously and conduct name translation driven decoding with language model (LM) based selection (Section 3.2).
4. Propose a new MT evaluation metric which can discriminate names and non-informative words (Section 4).

## 2 Baseline MT

As our baseline, we apply a high-performing Chinese-English MT system (Zheng, 2008; Zheng et al., 2009) based on hierarchical phrase-based translation framework (Chiang, 2005). It is based on a weighted synchronous context-free grammar (SCFG). All SCFG rules are associated with a set of features that are used to compute derivation probabilities. The features include:

- Relative frequency in two directions  $P(\gamma|\alpha)$  and  $P(\alpha|\gamma)$ , estimating the likelihoods of one side of the rule  $r: X \rightarrow \langle \gamma, \alpha \rangle$  translating into the other side, where  $\gamma$  and  $\alpha$  are strings of terminals and non-terminals in the source side and target side. Non-terminals in  $\gamma$  and  $\alpha$  are in one-to-one correspondence.
- Lexical weights in two directions:  $P_w(\gamma|\alpha)$  and  $P_w(\alpha|\gamma)$ , estimating likelihoods of words in one side of the rule  $r: X \rightarrow \langle \gamma, \alpha \rangle$  translating into the other side (Koehn et al., 2003).
- Phrase penalty: a penalty  $\exp(1)$  for a rule with no non-terminal being used in derivation.
- Rule penalty: a penalty  $\exp(1)$  for a rule with at least one non-terminal being used in derivation.
- Glue rule penalty: a penalty  $\exp(1)$  if a glue rule used in derivation.
- Translation length: number of words in translation output.

Our previous work showed that combining multiple LMs trained from different sources can lead to significant improvement. The LM used for decoding is a log-linear combination of four word n-gram LMs which are built on different English

corpora (details described in section 5.1), with the LM weights optimized on a development set and determined by minimum error rate training (MERT), to estimate the probability of a word given the preceding words. All four LMs were trained using modified Kneser-Ney smoothing algorithm (Chen and Goodman, 1996) and converted into Bloom filter LMs (Talbot and Brants, 2008) supporting memory map.

The scaling factors for all features are optimized by minimum error rate training algorithm to maximize BLEU score (Och, 2003). Given an input sentence in the source language, translation into the target language is cast as a search problem, where the goal is to find the highest-probability derivation that generates the source-side sentence, using the rules in our SCFG. The source-side derivation corresponds to a synchronous target-side derivation and the terminal yield of this target-side derivation is the output of the system. We employ our CKY-style chart decoder, named SRInterp, to solve the search problem.

## 3 Name-aware MT

We tightly integrate name processing into the above baseline to construct a NAMT model. Figure 1 depicts the general procedure.

### 3.1 Training

This basic training process of NAMT requires us to apply a bilingual name tagger to annotate parallel training corpora. Traditional name tagging approaches for single languages cannot address this requirement because they were all built on data and resources which are specific to each language without using any cross-lingual features. In addition, due to separate decoding processes the results on parallel data may not be consistent across languages. We developed a bilingual joint name tagger (Li et al., 2012) based on conditional random fields that incorporates both monolingual and cross-lingual features and conducts joint inference, so that name tagging from two languages can mutually enhance each other and therefore inconsistent results can be corrected simultaneously. This joint name tagger achieved 86.3% bilingual pair F-measure with manual alignment and 84.4% bilingual pair F-measure with automatic alignment as reported in (Li et al., 2012). Given a parallel sentence pair we first apply Giza++ (Och and Ney, 2003) to align words, and apply this joint

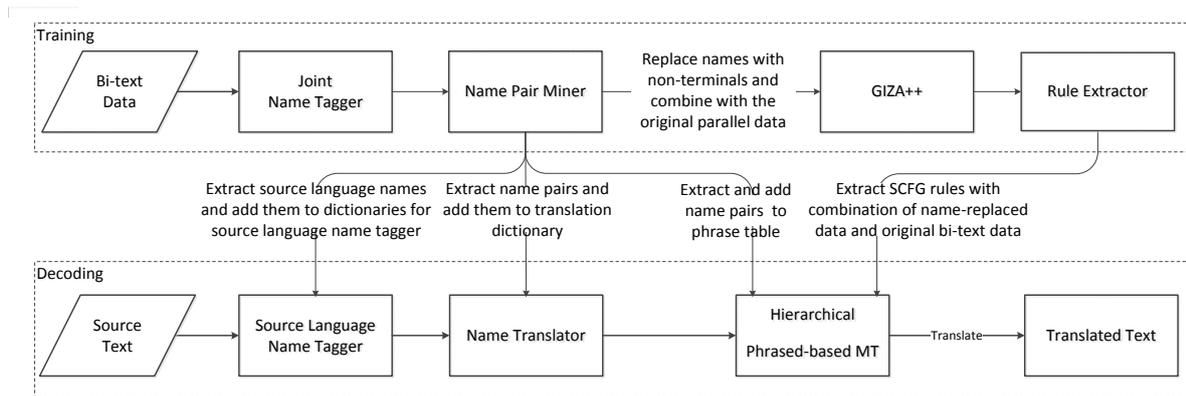


Figure 1: Architecture of Name-aware Machine Translation System.

t bilingual name tagger to extract three types of names: (Person (PER), Organization (ORG) and Geo-political entities (GPE)) from both the source side and the target side. We pair two entities from two languages, if they have the same entity type and are mapped together by word alignment. We ignore two kinds of names: multi-word names with conflicting boundaries in two languages and names only identified in one side of a parallel sentence.

We built a NAMT system from such name-tagged parallel corpora. First, we replace tagged name pairs with their entity types, and then use Giza++ and symmetrization heuristics to regenerate word alignment. Since the name tags appear very frequently, the existence of such tags yields improvement in word alignment quality. The re-aligned parallel corpora are used to train our NAMT system based on SCFG. Since the joint name tagger ensures that each tagged source name has a corresponding translation on the target side (and vice versa), we can extract SCFG rules by treating the tagged names as non-terminals.

However, the original parallel corpora contain many high-frequency names, which can already be handled well by the baseline MT. Some of these names carry special meanings that may influence translations of the neighboring words, and thus replacing them with non-terminals can lead to information loss and weaken the translation model. To address this issue, we merged the name-replaced parallel data with the original parallel data and extract grammars from the combined corpus. For example, given the following sentence pair:

- 中国 反对 外来 势力 介入 安哥拉 冲突 .
- China appeals to world for non involvement in Angola conflict .

after name tagging it becomes

- GPE 反对 外来 势力 介入 GPE 冲突 .
- GPE appeals to world for non involvement in GPE conflict .

Both sentence pairs are kept in the combined data to build the translation model.

### 3.2 Decoding

During decoding phase, we extract names with the baseline monolingual name tagger described in (Li et al., 2012) from a source document. Its performance is comparable to the best reported results on Chinese name tagging on Automatic Content Extraction (ACE) data (Ji and Grishman, 2006; Florian et al., 2006; Zitouni and Florian, 2008; Nguyen et al., 2010). Then we apply a state-of-the-art name translation system (Ji et al., 2009) to translate names into the target language. The name translation system is composed of the following steps: (1) Dictionary matching based on 150,041 name translation pairs; (2) Statistical name transliteration based on a structured perceptron model and a character based MT model (Dayne and Shahram, 2007); (3) Context information extraction based re-ranking.

In our NAMT framework, we add the following extensions to name translation.

We developed a name origin classifier based on Chinese last name list (446 name characters) and name structure parsing features to distinguish Chinese person names and foreign person names (Ji, 2009), so that pinyin conversion is applied for Chinese names while name transliteration is applied only for foreign names. This classifier works reasonably well in most cases (about 92% classification accuracy), except when a common Chinese last name appears as the first character of a foreign

name, such as “朱莉” which can be translated either as “Jolie” or “Zhu Li”.

For those names with fewer than five instances in the training data, we use the name translation system to provide translations; for the rest of the names, we leave them to the baseline MT model to handle. The joint bilingual name tagger was also exploited to mine bilingual name translation pairs from parallel training corpora. The mapping score between a Chinese name and an English name was computed by the number of aligned tokens. A name pair is extracted if the mapping score is the highest among all combinations and the name types on both sides are identical. It is necessary to incorporate word alignment as additional constraints because the order of names is often changed after translation. Finally, the extracted 9,963 unique name translation pairs were also used to create an additional name phrase table for NAMT. Manual evaluation on 2,000 name pairs showed the accuracy is 86%.

The non-terminals in SCFG rules are rewritten to the extracted names during decoding, therefore allow unseen names in the test data to be translated. Finally, based on LMs, our decoder exploits the dynamically created phrase table from name translation, competing with originally extracted rules, to find the best translation for the input sentence.

#### 4 Name-aware MT Evaluation

Traditional MT evaluation metrics such as BLEU (Papineni et al., 2002) and Translation Edit Rate (TER) (Snover et al., 2006) assign the same weights to all tokens equally. For example, incorrect translations of “the” and “Bush” will receive the same penalty. However, for cross-lingual information processing applications, we should acknowledge that certain informationally critical words are more important than other common words. In order to properly evaluate the translation quality of NAMT methods, we propose to modify the BLEU metric so that they can dynamically assign more weights to names during evaluation.

BLEU considers the correspondence between a system translation and a human translation:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^N w_n \log p_n\right) \quad (1)$$

where  $BP$  is brevity penalty defined as follows:

$$BP = \begin{cases} 1 & \text{if } c > r, \\ e^{(1-r/c)} & \text{if } c \leq r. \end{cases} \quad (2)$$

where  $w_n$  is a set of positive weights summing to one and usually uniformly set as  $w_n = 1/N$ ,  $c$  is the length of the system translation and  $r$  is the length of reference translation, and  $p_n$  is modified n-gram precision defined as:

$$p_n = \frac{\sum_{C \in \text{Candidates}} \sum_{\text{n-gram} \in C} \text{Count}_{clip}(\text{n-gram})}{\sum_{C' \in \text{Candidates}} \sum_{\text{n-gram}' \in C'} \text{Count}_{clip}(\text{n-gram}')} \quad (3)$$

where  $C$  and  $C'$  are translation candidates in the candidate sentence set, if a source sentence is translated to many candidate sentences.

As in BLEU metric, we first count the maximum number of times an n-gram occurs in any single reference translation. The total count of each candidate n-gram is clipped at sentence level by its maximum reference count. Then we add up the weights of clipped n-grams and divide them by the total weight of all n-grams.

Based on BLEU score, we design a name-aware BLEU metric as follows. Depending on whether a token  $t$  is contained in a name in reference translation, we assign a weight  $weight_t$  to  $t$  as follows:

$$weight_t = \begin{cases} 1 - e^{-tf(t,d) \cdot idf(t,D)}, & \text{if } t \text{ never appears in names} \\ 1 + \frac{PE}{Z}, & \text{if } t \text{ occurs in name(s)} \end{cases} \quad (4)$$

where  $PE$  is the sum of penalties of non-name tokens and  $Z$  is the number of tokens within all names:

$$PE = \sum_{t \text{ never appears in names}} e^{-tf(t,d) \cdot idf(t,D)} \quad (5)$$

In this paper, the  $tf \cdot idf$  score is computed at sentence level, therefore,  $D$  is the sentence set and each  $d \in D$  is a sentence.

The weight of an n-gram in reference translation is the sum of weights of all tokens it contains.

$$weight_{ngram} = \sum_{t \in ngram} weight_t \quad (6)$$

Next, we compute the weighted modified n-gram precision  $Count_{weight-clip}(\text{n-gram})$  as follows:

$$Count_{weight-clip}(\text{n-gram}) = \sum_{\text{if the } ngram_i \text{ is correctly translated}} weight_{ngram_i} \quad (7)$$

The  $Count_{clip}(n\text{-gram})$  in the equation 3 is substituted with above  $Count_{weight-clip}(n\text{-gram})$ . When we sum up the total weight of all n-grams of a candidate translation, some n-grams may contain tokens which do not exist in reference translation. We assign the lowest weight of tokens in reference translation to these rare tokens.

We also add an item, name penalty  $NP$ , to penalize the output sentences which contain too many or too few names:

$$NP = e^{-\left(\frac{u}{v}-1\right)^2/2\sigma} \quad (8)$$

where  $u$  is the number of name tokens in system translation and  $v$  is the number of name tokens in reference translation.

Finally the name-aware BLEU score is defined as:

$$BLEU_{NA} = BP \cdot NP \cdot \exp\left(\sum_{n=1}^N w_n \log wp_n\right) \quad (9)$$

This new metric can also be applied to evaluate MT approaches which emphasize other types of facts such as events, by simply replacing name tokens by other fact tokens.

## 5 Experiments

In this section we present the experimental results of NAMT compared to the baseline MT.

### 5.1 Data Set

We used a large Chinese-English MT training corpus from various sources and genres (including newswire, web text, broadcast news and broadcast conversations) for our experiments. We also used some translation lexicon data and Wikipedia translations. The majority of the data sets were collected or made available by LDC for U.S. DARPA Translingual Information Detection, Extraction and Summarization (TIDES) program, Global Autonomous Language Exploitation (GALE) program, Broad Operational Language Translation (BOLT) program and National Institute of Standards and Technology (NIST) MT evaluations. The training corpus includes 1,686,458 sentence pairs. The joint name tagger extracted 1,890,335 name pairs (295,087 Persons, 1,269,056 Geopolitical entities and 326,192 Organizations).

Four LMs, denoted LM1, LM2, LM3, and LM4, were trained from different English corpora. LM1 is a 7-gram LM trained on the tar-

get side of Chinese-English and Egyptian Arabic-English parallel text, English monolingual discussion forums data R1-R4 released in BOLT Phase 1 (LDC2012E04, LDC2012E16, LDC2012E21, LDC2012E54), and English Gigaword Fifth Edition (LDC2011T07). LM2 is a 7-gram LM trained only on the English monolingual discussion forums data listed above. LM3 is a 4-gram LM trained on the web genre among the target side of all parallel text (i.e., web text from pre-BOLT parallel text and BOLT released discussion forum parallel text). LM4 is a 4-gram LM trained on the English broadcast news and conversation transcripts released under the DARPA GALE program. Note that for LM4 training data, some transcripts were quick transcripts and quick rich transcripts released by LDC, and some were generated by running flexible alignment of closed captions or speech recognition output from LDC on the audio data (Venkataraman et al., 2004).

In order to demonstrate the effectiveness and generality of our approach, we evaluated our approach on seven test sets from multiple genres and domains. We asked four annotators to annotate names in four reference translations of each sentence and an expert annotator to adjudicate results. The detailed statistics and name distribution of each test data set is shown in Table 1. The percentage of names occurred fewer than 5 times in training data are listed in the brackets in the last column of the table.

### 5.2 Overall Performance

Besides the new name-aware MT metric, we also adopt two traditional metrics, TER to evaluate the overall translation performance and Named Entity Weak Accuracy (NEWA) (Hermjakob et al., 2008) to evaluate the name translation performance.

TER measures the amount of edits required to change a system output into one of the reference translations. Specifically:

$$TER = \frac{\# \text{ of edits}}{\text{average } \# \text{ of reference words}} \quad (10)$$

Possible edits include insertion, substitution deletion and shifts of words.

The NEWA metric is defined as follows. Using a manually assembled name variant table, we also support the matching of name variants (e.g., “World Health Organization” and “WHO”).

$$NEWA = \frac{\text{Count } \# \text{ of correctly translated names}}{\text{Count } \# \text{ of names in references}} \quad (11)$$

Corpus	Genre	Sentence #	Word # in source	Token # in reference	GPE(%)	PER(%)	ORG(%)	All names (% occurred < 5)
BOLT 1	forum	1,200	20,968	24,193	875(82.9)	90(8.5)	91(8.6)	1,056 (51.4)
BOLT 2	forum	1,283	23,707	25,759	815(73.7)	141(12.8)	149(13.5)	1,105 (65.9)
BOLT 3	forum	2,000	38,595	42,519	1,664(80.4)	204(9.8)	204(9.8)	2,072 (47.4)
BOLT 4	forum	1,918	41,759	47,755	1,852(80.0)	348(25.0)	113(5.0)	2,313 (53.3)
BOLT 5	blog	950	23,930	26,875	352(42.5)	235(28.3)	242(29.2)	829 (55.3)
NIST2006	news&blog	1,664	38,442	45,914	1,660(58.2)	568(19.9)	625(21.9)	2,853 (73.1)
NIST2008	news&blog	1,357	32,646	37,315	700(47.9)	367(25.1)	395(27.0)	1,462 (72.0)

**Table 1:** Statistics and Name Distribution of Test Data Sets.

Metric		System	BOLT 1	BOLT 2	BOLT 3	BOLT 4	BOLT 5	NIST2006	NIST2008
BLEU		Baseline	14.2	14.0	17.3	15.6	15.3	35.5	29.3
		NPhrase	14.1	14.4	17.1	15.4	15.3	35.4	29.3
		NAMT	14.2	<b>14.6</b>	16.9	<b>15.7</b>	<b>15.5</b>	<b>36.3</b>	<b>30.0</b>
Name-aware BLEU		Baseline	18.2	17.9	18.6	17.6	18.3	36.1	31.7
		NPhrase	18.1	18.8	18.5	18.1	18.0	35.8	31.8
		NAMT	<b>18.4</b>	<b>19.5</b>	<b>19.7</b>	<b>18.2</b>	<b>18.9</b>	<b>39.4</b>	<b>33.1</b>
TER		Baseline	70.6	71.0	69.4	70.3	67.1	58.7	61.0
		NPhrase	70.6	70.4	69.4	70.4	67.1	58.7	60.9
		NAMT	<b>70.3</b>	<b>70.2</b>	<b>69.2</b>	<b>70.1</b>	<b>66.6</b>	<b>57.7</b>	<b>60.5</b>
NEWA	All	Baseline	69.7	70.1	73.9	72.3	60.6	66.5	60.4
		NPhrase	69.8	71.1	73.8	72.5	60.6	68.3	61.9
		NAMT	<b>71.4</b>	<b>72.0</b>	<b>77.7</b>	<b>75.1</b>	<b>62.7</b>	<b>72.9</b>	<b>63.2</b>
	GPE	Baseline	72.8	78.4	80.0	78.7	81.3	79.2	76.0
		NPhrase	73.6	79.3	79.2	78.9	82.3	82.6	79.5
		NAMT	<b>74.2</b>	<b>80.2</b>	<b>82.8</b>	<b>80.4</b>	79.3	<b>85.5</b>	79.3
	PER	Baseline	53.3	44.7	45.1	49.4	48.9	54.2	51.2
		NPhrase	52.2	45.4	48.9	48.5	47.6	55.1	50.9
		NAMT	<b>55.6</b>	<b>45.4</b>	<b>58.8</b>	<b>55.2</b>	<b>56.2</b>	<b>60.0</b>	<b>52.3</b>
	ORG	Baseline	56.0	49.0	52.9	38.1	41.7	44.0	41.3
		NPhrase	50.5	50.3	54.4	40.7	41.3	42.2	40.7
		NAMT	<b>60.4</b>	<b>52.3</b>	<b>55.4</b>	<b>41.6</b>	<b>45.0</b>	<b>51.0</b>	<b>44.8</b>

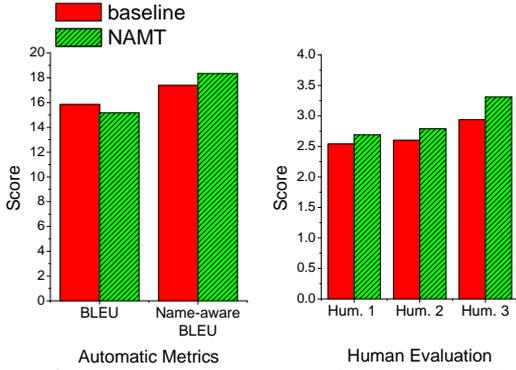
**Table 2:** Translation Performance (%).

For better comparison with NAMT, besides the original baseline, we develop the other baseline system by adding name translation table into the phrase table (NPhrase).

Table 2 presents the performance of overall translation and name translation. We can see that except for the BOLT3 data set with BLEU metric, our NAMT approach consistently outperformed the baseline system for all data sets with all metrics, and provided up to 23.6% relative error reduction on name translation. According to Wilcoxon Matched-Pairs Signed-Ranks Test, the improvement is not significant with BLEU metric, but is significant at 98% confidence level with all of the other metrics. The gains are more significant for formal genres than informal genres mainly because most of the training data for name tagging and name translation were from newswire. Furthermore, using external name translation table only did not improve translation quality in most test sets except for BOLT2. Therefore, it is important to use name-replaced corpora for rule extraction to fully take advantage of improved word alignment.

Many errors from the baseline MT approach oc-

curred because some parts of out-of-vocabulary names were mistakenly segmented into common words. For example, the baseline MT system mistakenly translated a person name “孙红雷 (*Sun Honglei*)” into “*Sun red thunder*”. In informal genres such as discussion forums and web blogs, even common names often appear in rare forms due to misspelling or morphing. For example, “奥巴马 (*Obama*)” was mistakenly translated into “*Ma Olympic*”. Such errors can be compounded when word re-ordering was applied. For example, the following sentence: “郭美美的力量还真是强大啊，真是佩服她 (*Guo Meimei’s strength really is formidable, I really admire her*)” was mistakenly translated into “*Guo the strength of the America and the America also really strong, ah, really admire her*” by the baseline MT system because the person name “郭美美 (*Guomeimei*)” was mistakenly segmented into three words “郭 (*Guo*)”, “美 (*the America*)” and “美 (*the America*)”. But our NAMT approach successfully identified and translated this name and also generated better overall translation: “*Guo Meimei’s power is also really strong, ah, really admire her*”.



**Figure 2:** Scores based on Automatic Metrics and Human Evaluation.

### 5.3 Name-aware BLEU vs The Human Evaluation

In order to investigate the correlation between name-aware BLEU scores and human judgment results, we asked three bi-lingual speakers to judge our translation output from the baseline system and the NAMT system, on a Chinese subset of 250 sentences (each sentence has two corresponding translations from baseline and NAMT) extracted randomly from 7 test corpora. The annotators rated each translation from 1 (very bad) to 5 (very good) and made their judgments based on whether the translation is understandable and conveys the same meaning.

We computed the name-aware BLEU scores on the subset and also the aggregated average scores from human judgments. Figure 2 shows that NAMT consistently achieved higher scores with both name-aware BLEU metric and human judgment. Furthermore, we calculated three Pearson product-moment correlation coefficients between human judgment scores and name-aware BLEU scores of these two MT systems. Give the sample size and the correlation coefficient value, the high significance value of 0.99 indicates that name-aware BLEU tracks human judgment well.

### 5.4 Word Alignment

It is also important to investigate the impact of our NAMT approach on improving word alignment. We conducted the experiment on the Chinese-English Parallel Treebank (Li et al., 2010) with ground-truth word alignment. The detailed procedure following NAMT framework is as follows: (1) Ran the joint bilingual name tagger; (2) Replaced each name string with its name type (PER, ORG or GPE), and ran Giza++ on the replaced sentences; (3) Ran Giza++ on the words within

Words	Method	P	R	F
Overall Words	Baseline Giza++	69.8	47.8	56.7
	Joint Name Tagging	<b>70.4</b>	<b>48.1</b>	<b>57.1</b>
	Ground-truth Name Tagging (Upper-bound)	71.3	48.9	58.0
Words Within Names	Baseline Giza++	86.0	31.4	46.0
	Joint Name Tagging	<b>77.6</b>	<b>37.2</b>	<b>50.3</b>

**Table 3:** Impact of Joint Bilingual Name Tagging on Word Alignment (%).

each name pair. (4) Merged (2) and (3) to produce the final word alignment results. In order to compare with the upper-bound gains, we also measured the performance of applying ground-truth name tagging with the above procedures.

The experiment results are shown in Table 3. For the words within names, our approach provided significant gains by enhancing F-measure from 46.0% to 50.3%. Only 10.6% words are within names, therefore the upper-bound gains on overall word alignment is only 1.3%. Our joint name tagging approach achieved 0.4% (statistically significant) improvement over the baseline. In Figure 3 we categorized the sentences according to the percentage of name words in each sentence and measured the improvement for each category. We can clearly see that as the sentences include more names, the gains achieved by our approach tend to be greater.

### 5.5 Remaining Error Analysis

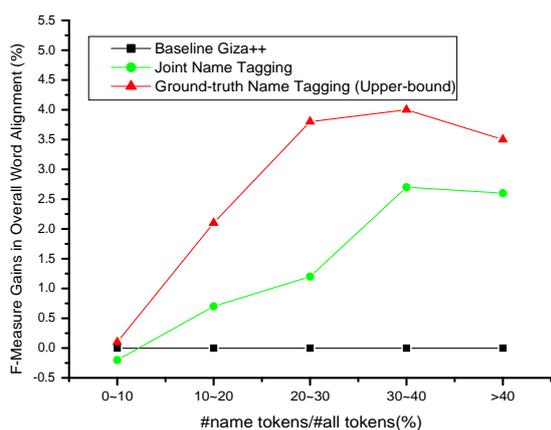
Although the proposed model has significantly enhanced translation quality, some challenges remain. We analyze some major sources of the remaining errors as follows.

#### 1. Name Structure Parsing.

We found that the gains of our NAMT approach were mainly achieved for names with one or two components. When the name structure becomes too complicated to parse, name tagging and name translation are likely to produce errors, especially for long nested organizations. For example, “古田县 检察院 反渎局” (Anti-malfeasance Bureau of Gutian County Procuratorate) consists of a nested organization name with a GPE as modifier: “古田县 检察院” (Gutian County Procuratorate) and an ORG name: “反渎局” (Anti-malfeasance Bureau).

#### 2. Name abbreviation tagging and translation.

Some organization abbreviations are also difficult to extract because our name taggers have



**Figure 3:** Word alignment gains according to the percentage of name words in each sentence.

not incorporated any coreference resolution techniques. For example, without knowing that “FAW” refers to “*First Automotive Works*” in “*FAW has also utilized the capital market to directly finance, and now owns three domestic listed companies*”, our system mistakenly labeled it as a GPE. The same challenge exists in name alignment and translation (for example, “民革 (Min Ge)” refers to “*中国国民党革命委员会*” (Revolutionary Committee of the Chinese Kuomintang).

### 3. Cross-lingual information transfer

English monolingual features normally generate higher confidence than Chinese features for ORG names. On the other hand, some good propagated Chinese features were not able to correct English results. For example, in the following sentence pair: “根据中国，老挝和联合国难民署三方达成的... (*in accordance with the tripartite agreement reached by China, Laos and the UNHCR on*)...”, even though the tagger can successfully label “联合国难民署/UNHCR” as an organization because it is a common Chinese name, English features based on previous GPE contexts still incorrectly predicted “UNHCR” as a GPE name.

## 6 Related Work

Two types of humble strategies were previously attempted to build name translation components which operate in tandem and loosely integrate into conventional statistical MT systems:

1. *Pre-processing*: identify names in the source texts and propose name translations to the MT system; the name translation results can be simply but aggressively transferred from the source to the target side using word alignment, or added into phrase table in order to

enable the LM to decide which translations to choose when encountering the names in the texts (Ji et al., 2009). Heuristic rules or supervised models can be developed to create “do-not-translate” list (Babych and Hartley, 2003) or learn “when-to-transliterate” (Hermjakob et al., 2008).

2. *Post-processing*: in a cross-lingual information retrieval or question answering framework, online query names can be utilized to obtain translation and post-edit MT output (Parton et al., 2009; Ma and McKeown, 2009; Parton and McKeown, 2010; Parton et al., 2012).

It is challenging to decide when to use name translation results. The simple transfer method ensures all name translations appear in the MT output, but it heavily relies on word alignment and does not take into account word re-ordering or the words found in a name’s context; therefore it could mistakenly break some context phrase structures due to name translation or alignment errors. The LM selection method often assigns an inappropriate weight to the additional name translation table because it is constructed independently from translation of context words; therefore after weighted voting most correct name translations are not used in the final translation output. Our experimental results 2 confirmed this weakness. More importantly, in these approaches the MT model was still mostly treated as a “black-box” because neither the translation model nor the LM was updated or adapted specifically for names.

Recently the wider idea of incorporating semantics into MT has received increased interests. Most of them designed some certain semantic representations, such as predicate-argument structure or semantic role labeling (Wu and Fung, 2009; Liu and Gildea, 2009; Meyer et al., 2011; Bojar and Wu, 2012), word sense disambiguation (Carpuat and Wu, 2007b; Carpuat and Wu, 2007a) and graph-structured grammar representation (Jones et al., 2012). Lo et al. (2012) proposed a semantic role driven MT metric. However, none of these work declaratively exploited results from information extraction for MT.

Some statistical MT systems (e.g. (Zens et al., 2005), (Aswani and Gaizauskas, 2005)) have attempted to use text normalization to improve word alignment for dates, numbers and job titles. But little reported work has shown the impact of joint

name tagging on overall word alignment.

Most of the previous name translation work combined supervised transliteration approaches with LM based re-scoring (Knight and Graehl, 1998; Al-Onaizan and Knight, 2002; Huang et al., 2004). Some recent research used comparable corpora to mine name translation pairs (Feng et al., 2004; Kutsumi et al., 2004; Udupa et al., 2009; Ji, 2009; Fung and Yee, 1998; Rapp, 1999; Shao and Ng, 2004; Lu and Zhao, 2006; Hassan et al., 2007). However, most of these approaches required large amount of seeds, suffered from Information Extraction errors, and relied on phonetic similarity, context co-occurrence and document similarity for re-scoring. In contrast, our name pair mining approach described in this paper does not require any machine translation or transliteration features.

## 7 Conclusions and Future Work

We developed a name-aware MT framework which tightly integrates name tagging and name translation into training and decoding of MT. Experiments on Chinese-English translation demonstrated the effectiveness of our approach over a high-quality MT baseline in both overall translation and name translation, especially for formal genres. We also proposed a new name-aware evaluation metric. In the future we intend to improve the framework by training a discriminative model to automatically assign weights to combine name translation and baseline translation with additional features including name confidence values, name types and global validation evidence, as well as conducting LM adaptation through bilingual topic modeling and clustering based on name annotations. We also plan to jointly optimize MT and name tagging by propagating multiple word segmentation and name annotation hypotheses in lattice structure to statistical MT and conduct lattice-based decoding (Dyer et al., 2008). Furthermore, we are interested in extending this framework to translate other out-of-vocabulary terms.

## Acknowledgement

This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. NSF CAREER Award under Grant IIS-0953149, the U.S. NSF EAGER Award under Grant No. IIS-1144111, the U.S. DARPA FA8750-13-2-0041 -

Deep Exploration and Filtering of Text (DEFT) Program and CUNY Junior Faculty Award. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on. We express our gratitude to Bing Zhao who provided the test sets and references that were used for Broad Operational Language Translation (BOLT) evaluation and thanks to Taylor Cassidy for constructive comments.

## References

- Y. Al-Onaizan and K. Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. In *Proceeding ACL'02*, pages 400–408.
- N. Aswani and R. Gaizauskas. 2005. A Hybrid Approach to Align Sentences and Words in English-Hindi Parallel Corpora. In *Proceeding ACL'05 Workshop on Building and Using Parallel Texts*, pages 57–64.
- Bogdan Babych and Anthony Hartley. 2003. Improving Machine Translation Quality with Automatic Named Entity Recognition. In *Proceeding EAMT '03 workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT*, pages 1–8.
- O. Bojar and D. Wu. 2012. Towards a Predicate-Argument evaluation for MT. In *Proceeding of the Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 30–38, July.
- Marine Carpuat and Dekai Wu. 2007a. How Phrase Sense Disambiguation outperforms Word Sense Disambiguation for Statistical Machine Translation. In *Proceeding TMI'07*, pages 43–52.
- Marine Carpuat and Dekai Wu. 2007b. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Proceeding EMNLP-CoNLL'07*, pages 61–72.
- Taylor Cassidy, Heng Ji, Hongbo Deng, Jing Zheng, and Jiawei Han. 2012. Analysis and Refinement of Cross-lingual Entity Linking. In *Proceeding CLEF'12*, pages 1–12.
- Stanley F. Chen and Joshua Goodman. 1996. An Empirical Study of Smoothing Techniques for Language Modeling. *Proceeding of ACL'96*, pages 310–318.

- David Chiang. 2005. A Hierarchical Phrase-based Model for Statistical Machine Translation. In *Proceeding ACL'05*, pages 263–270.
- F. Dayne and K. Shahram. 2007. A Sequence Alignment Model Based on the Averaged Perceptron. In *Proceeding EMNLP-CoNLL'07*, pages 238–247.
- C. Dyer, S. Muresan, and P. Resnik. 2008. Generalizing Word Lattice Translation. In *Proceeding ACL-HLT'08*, pages 1012–1020.
- D. Feng, Y. Lv, and M. Zhou. 2004. A New Approach for English-Chinese Named Entity Alignment. In *Proceeding PACLIC'04*, pages 372–379.
- R. Florian, H. Jing, N. Kambhatla, and I. Zitouni. 2006. Factorizing Complex Models: A Case Study in Mention Detection. In *Proceeding COLING-ACL'06*, pages 473–480.
- P. Fung and L. Y. Yee. 1998. An IR Approach for Translating New Words from Nonparallel and Comparable Texts. In *Proceeding COLING-ACL'98*, pages 414–420.
- D. Hakkani-Tur, H. Ji, and R. Grishman. 2007. Using Information Extraction to Improve Cross-lingual Document Retrieval. In *Proceeding RANLP Workshop on Multi-source, Multilingual Information Extraction and Summarization*, pages 17–23.
- A. Hassan, H. Fahmy, and H. Hassan. 2007. Improving Named Entity Translation by Exploiting Comparable and Parallel Corpora. In *Proceeding RANLP'07*, pages 1–6.
- U. Hermjakob, K. Knight, and H. Daume III. 2008. Name Translation in Statistical Machine Translation: Learning When to Transliterate. In *Proceeding ACL'08*, pages 389–397.
- F. Huang, S. Vogel, and A. Waibel. 2004. Improving Named Entity Translation Combining Phonetic and Semantic Similarities. In *Proceeding HLT/NAACL'04*, pages 281–288.
- H. Ji and R. Grishman. 2006. Analysis and Repair of Name Tagger Errors. In *Proceeding COLING-ACL'06*, pages 420–427.
- H. Ji, R. Grishman, D. Freitag, M. Blume, J. Wang, S. Khadivi, R. Zens, and H. Ney. 2009. Name Extraction and Translation for Distillation. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*.
- H. Ji. 2009. Mining Name Translations from Comparable Corpora by Creating Bilingual Information Networks. In *Proceeding ACL-IJCNLP'09 workshop on Building and Using Comparable Corpora*, pages 34–37.
- B. Jones, J. Andreas, D. Bauer, K. M. Hermann, and K. Knight. 2012. Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *Proceeding COLING'12*, pages 1359–1376.
- K. Knight and J. Graehl. 1998. Machine Transliteration. In *Computational Linguistics*, volume 24, pages 599–612, Cambridge, MA, USA, December. MIT Press.
- P. Koehn, F. Josef Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proceeding HLT-NAACL'03*, pages 127–133.
- T. Kutsumi, T. Yoshimi, K. Kotani, and I. Sata. 2004. Integrated Use of Internal and External Evidence in The Alignment of Multi-Word Named Entities. In *Proceeding PACLIC'04*, pages 187–196.
- X. Li, S. Strassel, S. Grimes, S. Ismael, X. Ma, N. Ge, A. Bies, N. Xue, and M. Maamouri. 2010. Parallel Aligned Treebank Corpora at LDC: Methodology, Annotation and Integration. In *Workshop on Annotation and Exploitation of Parallel Corpora (AEPC)*.
- Q. Li, H. Li, H. Ji, W. Wang, J. Zheng, and F. Huang. 2012. Joint Bilingual Name Tagging for Parallel Corpora. In *Proceeding CIKM'12*, pages 1727–1731.
- D. Liu and D. Gildea. 2009. Semantic Role Features for Machine Translation. In *Proceeding COLING'09*, pages 716–724.
- C. Lo, A. K. Tumuluru, and D. Wu. 2012. Fully Automatic Semantic MT Evaluation. In *Proceeding of the Seventh Workshop on Statistical Machine Translation*, pages 243–252.
- M. Lu and J. Zhao. 2006. Multi-feature based Chinese-English Named Entity Extraction from Comparable Corpora. In *Proceeding PACLIC'06*, pages 134–141.
- W. Ma and K. McKeown. 2009. Where's the Verb Correcting Machine Translation During Question Answering. In *Proceeding ACL-IJCNLP'09*, pages 333–336.
- P. McNamee, J. Mayfield, D. Lawrie, D. W. Oard, and D. Doermann. 2011. Cross-Language Entity Linking. In *Proceeding IJCNLP'11*.
- A. Meyer, M. Kosaka, S. Liao, and N. Xue. 2011. Improving MT Word Alignment Using Aligned Multi-Stage Parses. In *Proceeding ACL-HLT 2011 Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 88–97.
- T. T. Nguyen, A. Moschitti, and G. Riccardi. 2010. Kernel-based Reranking for Named-Entity Extraction. In *Proceeding COLING'10*, pages 901–909.
- F. J. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

- F. J. Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceeding ACL'03*, pages 160–167.
- K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceeding ACL'02*, pages 311–318.
- K. Parton and K. McKeown. 2010. MT Error Detection for Cross-Lingual Question Answering. *Proceeding COLING'10*, pages 946–954.
- K. Parton, K. R. McKeown, R. Coyne, M. T. Diab, R. Grishman, D. Hakkani-Tur, M. Harper, H. Ji, W. Y. Ma, A. Meyers, S. Stolbach, A. Sun, G. Tur, W. Xu, and S. Yaman. 2009. Who, What, When, Where, Why? Comparing Multiple Approaches to the Cross-Lingual 5W Task. In *Proceeding ACL-IJCNLP'09*, pages 423–431.
- K. Parton, N. Habash, K. McKeown, G. Iglesias, and A. de Gispert. 2012. Can Automatic Post-Editing Make MT More Meaningful? In *Proceeding EAMT'12*, pages 111–118.
- R. Rapp. 1999. Automatic Identification of Word Translations from Unrelated English and German Corpora. In *Proceeding ACL'99*, pages 519–526.
- L. Shao and H. T. Ng. 2004. Mining New Word Translations from Comparable Corpora. In *Proceeding COLING'04*.
- M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceeding of Association for Machine Translation in the Americas*, pages 223–231.
- M. Snover, X. Li, W. Lin, Z. Chen, S. Tamang, M. Ge, A. Lee, Q. Li, H. Li, S. Anzaroot, and H. Ji. 2011. Cross-lingual Slot Filling from Comparable Corpora. In *Proceeding ACL'11 Workshop on Building and Using Comparable Corpora*, pages 110–119.
- D. Talbot and T. Brants. 2008. Randomized Language Models via Perfect Hash Functions. In *Proceeding of ACL/HLT'08*, pages 505–513.
- R. Udupa, K. Saravanan, A. Kumaran, and J. Jagarlamudi. 2009. MINT: A Method for Effective and Scalable Mining of Named Entity Transliterations from Large Comparable Corpora. In *Proceeding EACL'09*, pages 799–807.
- A. Venkataraman, A. Stolcke, W. Wang, D. Vergyri, V. R. R. Gadde, and J. Zheng. 2004. An Efficient Repair Procedure For Quick Transcriptions. In *Proceeding INTERSPEECH'04*, pages 1961–1964.
- D. Wu and P. Fung. 2009. Semantic Roles for SMT: A Hybrid Two-Pass Model. In *NAACL HLT'09*, pages 13–16.
- R. Zens, O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. 2005. The RWTH Phrase-based Statistical Machine Translation System. In *Proceeding IWSLT'05*, pages 155–162.
- J. Zheng, N. F. Ayan, W. Wang, and D. Burkett. 2009. Using Syntax in Large-Scale Audio Document Translation. In *Proceeding Interspeech'09*, pages 440–443.
- J. Zheng. 2008. SRInterp: SRI's Scalable Multipurpose SMT Engine. In *Technical Report*.
- I. Zitouni and R. Florian. 2008. Mention Detection Crossing the Language Barrier. In *Proceeding EMNLP'08*, pages 600–609.

# Decipherment Complexity in 1:1 Substitution Ciphers

Malte Nuhn and Hermann Ney

Human Language Technology and Pattern Recognition  
Computer Science Department, RWTH Aachen University, Aachen, Germany

<surname>@cs.rwth-aachen.de

## Abstract

In this paper we show that even for the case of 1:1 substitution ciphers—which encipher plaintext symbols by exchanging them with a unique substitute—finding the optimal decipherment with respect to a bigram language model is NP-hard. We show that in this case the decipherment problem is equivalent to the quadratic assignment problem (QAP). To the best of our knowledge, this connection between the QAP and the decipherment problem has not been known in the literature before.

## 1 Introduction

The decipherment approach for MT has recently gained popularity for training and adapting translation models using only monolingual data. The general idea is to find those translation model parameters that maximize the probability of the translations of a given source text in a given language model of the target language.

In general, the process of translation has a wide range of phenomena like substitution and reordering of words and phrases. In this paper we only study models that substitute tokens—i.e. words or letters—with a unique substitute. It therefore serves as a very basic case for decipherment and machine translation.

Multiple techniques like integer linear programming (ILP),  $A^*$  search, genetic algorithms, and Bayesian inference have been used to tackle the decipherment problem for 1:1 substitution ciphers. The existence of such a variety of different approaches for solving the same problem already shows that there is no obvious way to solve the problem optimally.

In this paper we show that decipherment of 1:1 substitution ciphers is indeed NP-hard and thus ex-

plain why there is no single best approach to the problem. The literature on decipherment provides surprisingly little on the analysis of the complexity of the decipherment problem. This might be related to the fact that a statistical formulation of the decipherment problem has not been analyzed with respect to  $n$ -gram language models: This paper shows the close relationship of the decipherment problem to the quadratic assignment problem. To the best of our knowledge the connection between the decipherment problem and the quadratic assignment problem was not known.

The remainder of this paper is structured as follows: In Section 2 we review related work. Section 3 introduces the decipherment problem and describes the notation and definitions used throughout this paper. In Section 4 we show that decipherment using a unigram language model corresponds to solving a linear sum assignment problem (LSAP). Section 5 shows the connection between the quadratic assignment problem and decipherment using a bigram language model. Here we also give a reduction of the traveling salesman problem (TSP) to the decipherment problem to highlight the additional complexity in the decipherment problem.

## 2 Related Work

In recent years a large number of publications on the automatic decipherment of substitution ciphers has been published. These publications were mostly dominated by rather heuristic methods and did not provide a theoretical analysis of the complexity of the decipherment problem: (Knight and Yamada, 1999) and (Knight et al., 2006) use the EM algorithm for various decipherment problems, like e.g. word substitution ciphers. (Ravi and Knight, 2008) and (Corlett and Penn, 2010) are able to obtain optimal (i.e. without search errors) decipherments of short cryptograms given an  $n$ -

gram language model. (Ravi and Knight, 2011), (Nuhn et al., 2012), and (Dou and Knight, 2012) treat natural language translation as a deciphering problem including phenomena like reordering, insertion, and deletion and are able to train translation models using only monolingual data.

In this paper we will show the connection between the decipherment problem and the linear sum assignment problem as well as the quadratic assignment problem: Regarding the linear sum assignment problem we will make use of definitions presented in (Burkard and el a, 1999). Concerning the quadratic assignment problem we will use basic definitions from (Beckmann and Koopmans, 1957). Further (Burkard et al., 1998) gives a good overview over the quadratic assignment problem, including different formulations, solution methods, and an analysis of computational complexity. The paper also references a vast amount of further literature that might be interesting for future research.

### 3 Definitions

In the following we will use the machine translation notation and denote the **ciphertext** with  $f_1^N = f_1 \dots f_j \dots f_N$  which consists of cipher tokens  $f_j \in V_f$ . We denote the **plaintext** with  $e_1^N = e_1 \dots e_i \dots e_N$  (and its vocabulary  $V_e$  respectively). We define

$$e_0 = f_0 = e_{N+1} = f_{N+1} = \$ \quad (1)$$

with “\$” being a special sentence boundary token. We use the abbreviations  $\bar{V}_e = V_e \cup \{\$\}$  and  $\bar{V}_f$  respectively.

A **general substitution cipher** uses a table  $s(e|f)$  which contains for each cipher token  $f$  a probability that the token  $f$  is substituted with the plaintext token  $e$ . Such a table for substituting cipher tokens  $\{A, B, C, D\}$  with plaintext tokens  $\{a, b, c, d\}$  could for example look like

	a	b	c	d
A	0.1	0.2	0.3	0.4
B	0.4	0.2	0.1	0.3
C	0.4	0.1	0.2	0.3
D	0.3	0.4	0.2	0.1

The **1:1 substitution cipher** encrypts a given plaintext into a ciphertext by replacing each plaintext token with a unique substitute: This means that the table  $s(e|f)$  contains all zeroes, except for

one “1.0” per  $f \in V_f$  and one “1.0” per  $e \in V_e$ . For example the text

abadcab

would be enciphered to

BCBADBC

when using the substitution

	a	b	c	d
A	0	0	0	1
B	1	0	0	0
C	0	1	0	0
D	0	0	1	0

We formalize the 1:1 substitutions with a bijective function  $\phi : V_f \rightarrow V_e$ . The general **decipherment goal** is to obtain a mapping  $\phi$  such that the probability of the deciphered text is maximal:

$$\hat{\phi} = \arg \max_{\phi} p(\phi(f_1)\phi(f_2)\phi(f_3)\dots\phi(f_N)) \quad (2)$$

Here  $p(\dots)$  denotes the **language model**. Depending on the structure of the language model Equation 2 can be further simplified.

Given a ciphertext  $f_1^N$ , we define the **unigram count**  $N_f$  of  $f \in \bar{V}_f$  as<sup>1</sup>

$$N_f = \sum_{i=0}^{N+1} \delta(f, f_i) \quad (3)$$

This implies that  $N_f$  are integer counts  $> 0$ . We similarly define the **bigram count**  $N_{ff'}$  of  $f, f' \in \bar{V}_f$  as

$$N_{ff'} = \sum_{i=1}^{N+1} \delta(f, f_{i-1}) \cdot \delta(f', f_i) \quad (4)$$

This definition implies that

- (a)  $N_{ff'}$  are integer counts  $> 0$  of bigrams found in the ciphertext  $f_1^N$ .
- (b) Given the first and last token of the cipher  $f_1$  and  $f_N$ , the bigram counts involving the sentence boundary token \$ need to fulfill

$$N_{\$f} = \delta(f, f_1) \quad (5)$$

$$N_{f\$} = \delta(f, f_N) \quad (6)$$

- (c) For all  $f \in V_f$

$$\sum_{f' \in V_f} N_{ff'} = \sum_{f' \in V_f} N_{f'f} \quad (7)$$

must hold.

<sup>1</sup>Here  $\delta$  denotes the Kronecker delta.

Similarly, we define language model matrices  $S$  for the unigram and the bigram case. The **unigram language model**  $S_f$  is defined as

$$S_f = \log p(f) \quad (8)$$

which implies that

(a)  $S_f$  are real numbers with

$$S_f \in [-\infty, 0] \quad (9)$$

(b) The following normalization constraint holds:

$$\sum_{f \in V_f} \exp(S_f) = 1 \quad (10)$$

Similarly for the **bigram language model matrix**  $S_{ff'}$ , we define

$$S_{ff'} = \log p(f'|f) \quad (11)$$

This definition implies that

(a)  $S_{ff'}$  are real numbers with

$$S_{ff'} \in [-\infty, 0] \quad (12)$$

(b) For the sentence boundary symbol, it holds that

$$S_{\S\S} = -\infty \quad (13)$$

(c) For all  $f \in \overline{V_f}$  the following normalization constraint holds:

$$\sum_{f' \in \overline{V_f}} \exp(S_{ff'}) = 1 \quad (14)$$

## 4 Decipherment Using Unigram LMs

### 4.1 Problem Definition

When using a unigram language model, Equation 2 simplifies to finding

$$\hat{\phi} = \arg \max_{\phi} \prod_{i=1}^N p(\phi(f_i)) \quad (15)$$

which can be rewritten as

$$\hat{\phi} = \arg \max_{\phi} \sum_{f \in V_f} N_f S_{\phi}(f) \quad (16)$$

When defining  $c_{ff'} = N_f \log p(f')$ , for  $f, f' \in V_f$ , Equation 16 can be brought into the form of

$$\hat{\phi} = \arg \max_{\phi} \sum_{f \in V_f} c_{f\phi}(f) \quad (17)$$

Figure 1 shows an illustration of this problem.

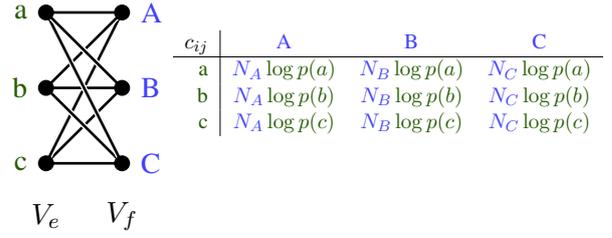


Figure 1: Linear sum assignment problem for a cipher with  $V_e = \{a, b, c\}$ ,  $V_f = \{A, B, C\}$ , unigram counts  $N_f$ , and unigram probabilities  $p(e)$ .

## 4.2 The Linear Sum Assignment Problem

The practical problem behind the linear sum assignment problem can be described as follows: Given jobs  $\{j_1, \dots, j_n\}$  and workers  $\{w_1, \dots, w_n\}$ , the task is to assign each job  $j_i$  to a worker  $w_j$ . Each assignment incurs a cost  $c_{ij}$  and the total cost for assigning all jobs and workers is to be minimized.

This can be formalized as finding the assignment

$$\hat{\phi} = \arg \min_{\phi} \sum_{i=1}^n c_{i\phi(i)} \quad (18)$$

The general LSAP can be solved in polynomial time using the Hungarian algorithm (Kuhn, 1955). However, since the matrix  $c_{ij}$  occurring for the decipherment using a unigram language model can be represented as the product  $c_{ij} = a_i \cdot b_j$  the decipherment problem can be solved more easily: In the Section “Optimal Matching”, (Bauer, 2010) shows that in this case the optimal assignment is found by sorting the jobs  $j_i$  by  $a_i$  and workers  $w_j$  by  $b_j$  and then assigning the jobs  $j_i$  to workers  $w_j$  that have the same rank in the respective sorted lists. Sorting and then assigning the elements can be done in  $\mathcal{O}(n \log n)$ .

## 5 Decipherment Using Bigram LMs

### 5.1 Problem Definition

When using a 2-gram language model, Equation 2 simplifies to

$$\hat{\phi} = \arg \max_{\phi} \left\{ \prod_{j=1}^{N+1} p(\phi(f_j) | \phi(f_{j-1})) \right\} \quad (19)$$

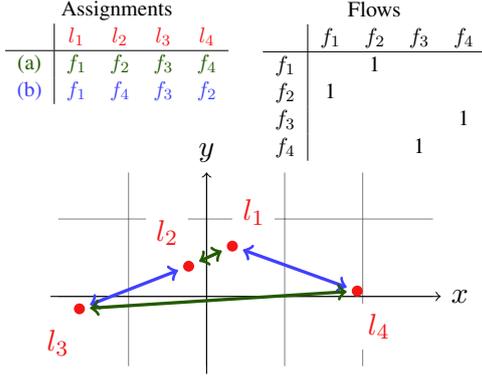


Figure 2: Hypothetical quadratic assignment problem with locations  $l_1 \dots l_4$  and facilities  $f_1 \dots f_4$  with all flows being zero except  $f_1 \leftrightarrow f_2$  and  $f_3 \leftrightarrow f_4$ . The distance between locations  $l_1 \dots l_4$  is implicitly given by the locations in the plane, implying a euclidean metric. Two example assignments (a) and (b) are shown, with (b) having the lower overall costs.

Using the definitions from Section 3, Equation 19 can be rewritten as

$$\hat{\phi} = \arg \max_{\phi} \left\{ \sum_{f \in V_f} \sum_{f' \in V_f} N_{ff'} S_{\phi(f)\phi(f')} \right\} \quad (20)$$

(Bauer, 2010) arrives at a similar optimization problem for the “combined method of frequency matching” using bigrams and mentions that it can be seen as a combinatorial problem for which an efficient way of solving is not known. However, he does not mention the close connection to the quadratic assignment problem.

## 5.2 The Quadratic Assignment Problem

The quadratic assignment problem was introduced by (Beckmann and Koopmans, 1957) for the following real-life problem:

Given a set of facilities  $\{f_1, \dots, f_n\}$  and a set of locations  $\{l_1, \dots, l_n\}$  with distances for each pair of locations, and flows for each pair of facilities (e.g. the amount of supplies to be transported between a pair of facilities) the problem is to assign the facilities to locations such that the sum of the distances multiplied by the corresponding flows (which can be interpreted as total transportation costs) is minimized. This is visualized in Figure 2.

Following (Beckmann and Koopmans, 1957) we can express the quadratic assignment problem

as finding

$$\hat{\phi} = \arg \min_{\phi} \left\{ \sum_{i=1}^n \sum_{j=1}^n a_{ij} b_{\phi(i)\phi(j)} + \sum_{i=1}^n c_{i\phi(i)} \right\} \quad (21)$$

where  $A = (a_{ij}), B = (b_{ij}), C = (c_{ij}) \in \mathbb{N}^{n \times n}$  and  $\phi$  a permutation

$$\phi : \{1, \dots, n\} \rightarrow \{1, \dots, n\}. \quad (22)$$

This formulation is often referred to as Koopman-Beckman QAP and often abbreviated as QAP( $A, B, C$ ). The so-called *pure* or *homogeneous* QAP

$$\hat{\phi} = \arg \min_{\phi} \left\{ \sum_{i=1}^n \sum_{j=1}^n a_{ij} b_{\phi(i)\phi(j)} \right\} \quad (23)$$

is obtained by setting  $c_{ij} = 0$ , and is often denoted as QAP( $A, B$ ).

In terms of the real-life problem presented in (Beckmann and Koopmans, 1957) the matrix  $A$  can be interpreted as distance matrix for locations  $\{l_1 \dots l_n\}$  and  $B$  as flow matrix for facilities  $\{f_1 \dots f_n\}$ .

(Sahni and Gonzalez, 1976) show that the quadratic assignment problem is strongly NP-hard.

We will now show the relation between the quadratic assignment problem and the decipherment problem.

### 5.3 Decipherment Problem $\preceq$ Quadratic Assignment Problem

Every decipherment problem is directly a quadratic assignment problem, since the matrices  $N_{ff'}$  and  $S_{ff'}$  are just special cases of the general matrices  $A$  and  $B$  required for the quadratic assignment problem. Thus a reduction from the decipherment problem to the quadratic assignment problem is trivial. This means that all algorithms capable of solving QAPs can directly be used to solve the decipherment problem.

### 5.4 Quadratic Assignment Problem $\preceq$ Decipherment Problem

Given QAP( $A, B$ ) with integer matrices  $A = (a_{ij}), B = (b_{ij}) i, j \in \{1, \dots, n\}$  we construct the count matrix  $N_{ff'}$  and language model matrix  $S_{ff'}$  in such a way that the solution for the decipherment problem implies the solution to the

quadratic assignment problem, and vice versa. We will use the vocabularies  $\bar{V}_e = \bar{V}_f = \{1, \dots, n+3\}$ , with  $n+3$  being the special sentence boundary token “\$”. The construction of  $N_{ff'}$  and  $S_{ff'}$  is shown in Figure 3.

To show the validity of our construction, we will

1. Show that  $N_{ff'}$  is a valid count matrix.
2. Show that  $S_{ff'}$  is a valid bigram language model matrix.
3. Show that the decipherment problem and the newly constructed quadratic assignment problem are equivalent.

We start by showing that  $N_{ff'}$  is a valid count matrix:

- (a) By construction,  $N_{ff'}$  has integer counts that are greater or equal to 0.
- (b) By construction,  $N_{ff'}$  at boundaries is:
  - $N_{f\$} = \delta(f, 1)$
  - $N_{f\$} = \delta(f, n+2)$

- (c) Regarding the properties  $\sum_{f'} N_{ff'} = \sum_{f'} N_{f'f}$ :

- For all  $f \in \{1, \dots, n\}$  the count properties are equivalent to

$$\tilde{a}_{f*} + \sum_{f'} \tilde{a}_{ff'} = \tilde{a}_{*f} + \sum_{f'} \tilde{a}_{f'f} + \delta(f, 1) \quad (24)$$

which holds by construction of  $\tilde{a}_{*f}$  and  $\tilde{a}_{f*}$ .

- For  $f = n+1$  the count property is equivalent to

$$1 + \sum_{f'} \tilde{a}_{f'*} = 2 + \sum_{f'} \tilde{a}_{*f'} \quad (25)$$

which follows from Equation (24) by summing over all  $f \in \{1, \dots, n\}$ .

- For  $f = n+2$  and  $f = n+3$ , the condition is fulfilled by construction.

We now show that  $S_{ff'}$  is a valid bigram language model matrix:

- (a) By construction,  $S_{ff'} \in [-\infty, 0]$  holds.
- (b) By construction,  $S_{\$\$} = -\infty$  holds.

- (c) By the construction of  $\tilde{b}_{f*}$ , the values  $S_{ff'}$  fulfill  $\sum_{f'} \exp(S_{ff'}) = 1$  for all  $f$ . This works since all entries  $\tilde{b}_{ff'}$  are chosen to be smaller than  $-\log(n+2)$ .

We now show the equivalence of the quadratic assignment problem and the newly constructed decipherment problem. For this we will use the definitions

$$\tilde{A} = \{1, \dots, n\} \quad (26)$$

$$\tilde{B} = \{n+1, n+2, n+3\} \quad (27)$$

We first show that solutions of the constructed decipherment problem with score  $> -\infty$  fulfill  $\phi(f) = f$  for  $f \in \tilde{B}$ .

All mappings  $\phi$ , with  $\phi(f) = f'$  for any  $f \in \tilde{A}$  and  $f' \in \tilde{B}$  will induce a score of  $-\infty$  since for  $f \in \tilde{A}$  all  $N_{ff} > 0$  and  $S_{ff'} = -\infty$  for  $f' \in \tilde{B}$ . Thus any  $\phi$  with score  $> -\infty$  will fulfill  $\phi(f) \in \tilde{B}$  for  $f \in \tilde{B}$ . Further, by enumerating all six possible permutations, it can be seen that only the  $\phi$  with  $\phi(f) = f$  for  $f \in \tilde{B}$  induces a score of  $> -\infty$ . Thus we can rewrite

$$\sum_{f=1}^{n+3} \sum_{f'=1}^{n+3} N_{ff'} S_{\phi(f)\phi(f')} \quad (28)$$

to

$$\underbrace{\sum_{f \in \tilde{A}} \sum_{f' \in \tilde{A}} N_{ff'} S_{\phi(f)\phi(f')}}_{(AA)} + \underbrace{\sum_{f \in \tilde{A}} \sum_{f' \in \tilde{B}} N_{ff'} S_{\phi(f)f'}}_{(AB)} + \underbrace{\sum_{f \in \tilde{B}} \sum_{f' \in \tilde{A}} N_{ff'} S_{f\phi(f')}}_{(BA)} + \underbrace{\sum_{f \in \tilde{B}} \sum_{f' \in \tilde{B}} N_{ff'} S_{ff'}}_{(BB)}$$

Here

- (AB) is independent of  $\phi$  since

$$\forall f \in \tilde{A}, f' \in \{n+1, n+2, n+3\} : S_{ff'} = S_{1f'} \quad (29)$$

and

$$\forall f \in \tilde{A} : N_{f,n+2} = 0 \quad (30)$$

- (BA) is independent of  $\phi$  since

$$\forall f' \in \tilde{A}, f \in \tilde{B} : S_{ff'} = S_{f1} \quad (31)$$

- (BB) is independent of  $\phi$ .

$$N_{ff'} = \left( \begin{array}{cccc|ccc} \tilde{a}_{11} & \tilde{a}_{12} & \cdots & \tilde{a}_{1n} & \tilde{a}_{1*} & 0 & 0 \\ \tilde{a}_{21} & \tilde{a}_{22} & \cdots & \tilde{a}_{2n} & \tilde{a}_{2*} & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \tilde{a}_{n1} & \tilde{a}_{n2} & \cdots & \tilde{a}_{nn} & \tilde{a}_{n*} & 0 & 0 \\ \hline \tilde{a}_{*1} & \tilde{a}_{*2} & \cdots & \tilde{a}_{*n} & 0 & 2 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & 1 \\ \hline 1 & 0 & \cdots & 0 & 0 & 0 & 0 \end{array} \right)$$

$$S_{ff'} = \left( \begin{array}{cccc|ccc} \tilde{b}_{11} & \tilde{b}_{12} & \cdots & \tilde{b}_{1n} & \varepsilon_2 & \tilde{b}_{1*} & \varepsilon_2 \\ \tilde{b}_{21} & \tilde{b}_{22} & \cdots & \tilde{b}_{2n} & \varepsilon_2 & \tilde{b}_{2*} & \varepsilon_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \tilde{b}_{n1} & \tilde{b}_{n2} & \cdots & \tilde{b}_{nn} & \varepsilon_2 & \tilde{b}_{n*} & \varepsilon_2 \\ \hline \varepsilon_1 & \varepsilon_1 & \cdots & \varepsilon_1 & -\infty & \varepsilon_1 & -\infty \\ \varepsilon_2 & \varepsilon_2 & \cdots & \varepsilon_2 & \varepsilon_2 & -\infty & \varepsilon_2 \\ \hline \varepsilon_0 & \varepsilon_0 & \cdots & \varepsilon_0 & -\infty & -\infty & -\infty \end{array} \right)$$

$$\tilde{a}_{ff'} = a_{ff'} - \min_{\tilde{f}\tilde{f}'} \{a_{\tilde{f}\tilde{f}'}\} + 1$$

$$\tilde{b}_{ff'} = b_{ff'} - \max_{\tilde{f}\tilde{f}'} \{b_{\tilde{f}\tilde{f}'}\} - \log(n+2)$$

$$\tilde{a}_{f*} = \max \left\{ \sum_{f'=1}^n a_{ff'} - a_{ff'}, 0 \right\} + \delta(f, 1)$$

$$\tilde{b}_{f*} = \log \left( 1 - \sum_{f'=1}^n \exp(\tilde{b}_{ff'}) - \frac{2}{n+2} \right)$$

$$\tilde{a}_{*f'} = \max \left\{ \sum_{f=1}^n a_{ff'} - a_{ff'}, 0 \right\}$$

$$\varepsilon_i = -\log(n+i)$$

Figure 3: Construction of matrices  $N_{ff'}$  and  $S_{ff'}$  of the decipherment problem from matrices  $A = (a_{ij})$  and  $B = (b_{ij})$  of the quadratic assignment problem  $QAP(A, B)$ .

Thus, with some constant  $c$ , we can finally rewrite Equation 28 as

$$c + \sum_{f=1}^n \sum_{f'=1}^n N_{ff'} S_{\phi(f)\phi(f')} \quad (32)$$

Inserting the definition of  $N_{ff'}$  and  $S_{ff'}$  (simplified using constants  $c'$ , and  $c''$ ) we obtain

$$c + \sum_{f=1}^n \sum_{f'=1}^n (a_{ff'} + c') (b_{\phi(f)\phi(f')} + c'') \quad (33)$$

which is equivalent to the original quadratic assignment problem

$$\arg \max \left\{ \sum_{f=1}^n \sum_{f'=1}^n a_{ff'} b_{\phi(f)\phi(f')} \right\} \quad (34)$$

Thus we have shown that a solution to the quadratic assignment problem in Equation 34 is a solution to the decipherment problem in Equation 20 and vice versa. Assuming that calculating elementary functions can be done in  $\mathcal{O}(1)$ , setting up  $N_{ff'}$  and  $S_{ff'}$  can be done in polynomial time.<sup>2</sup> Thus we have given a polynomial time reduction from the quadratic assignment problem to

<sup>2</sup>This is the case if we only require a fixed number of digits precision for the log and exp operations.

the decipherment problem: Since the quadratic assignment problem is NP-hard, it follows that the decipherment problem is NP-hard, too.

### 5.5 Traveling Salesman Problem $\preceq$ Decipherment Problem

Using the above construction we can immediately construct a decipherment problem that is equivalent to the traveling salesman problem by using the quadratic assignment problem formulation of the traveling salesman problem.

Without loss of generality<sup>3</sup> we assume that the TSP's distance matrix fulfills the constraints of a bigram language model matrix  $S_{ff'}$ . Then the count matrix  $N_{ff'}$  needs to be chosen as

$$N_{ff'} = \left( \begin{array}{ccccccc} 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \end{array} \right) \quad (35)$$

which fulfills the constraints of a bigram count matrix.

<sup>3</sup>The general case can be covered using the reduction shown in Section 5.

This matrix corresponds to a ciphertext of the form

$$\$abcd\$ \quad (36)$$

and represents the tour of the traveling salesman in an intuitive way. The mapping  $\phi$  then only decides in which order the cities are visited, and only costs between two successive cities are counted.

This shows that the TSP is only a special case of the decipherment problem.

## 6 Conclusion

We have shown the correspondence between solving 1:1 substitution ciphers and the linear sum assignment problem and the quadratic assignment problem: When using unigram language models, the decipherment problem is equivalent to the linear sum assignment problem and solvable in polynomial time. For a bigram language model, the decipherment problem is equivalent to the quadratic assignment problem and is NP-hard.

We also pointed out that all available algorithms for the quadratic assignment problem can be directly used to solve the decipherment problem.

To the best of our knowledge, this correspondence between the decipherment problem and the quadratic assignment problem has not been known previous to our work.

## Acknowledgements

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

## References

- Friedrich L. Bauer. 2010. *Decrypted Secrets: Methods and Maxims of Cryptology*. Springer, 4th edition.
- Martin J. Beckmann and Tjalling C. Koopmans. 1957. Assignment problems and the location of economic activities. *Econometrica*, 25(4):53–76.
- Rainer E. Burkard and Eranda ela. 1999. Linear assignment problems and extensions. In *Handbook of Combinatorial Optimization - Supplement Volume A*, pages 75–149. Kluwer Academic Publishers.
- Rainer E. Burkard, Eranda ela, Panos M. Pardalos, and Leonidas S. Pitsoulis. 1998. The quadratic assignment problem. In *Handbook of Combinatorial Optimization*, pages 241–338. Kluwer Academic Publishers.
- Eric Corlett and Gerald Penn. 2010. An exact A\* method for deciphering letter-substitution ciphers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1040–1047, Uppsala, Sweden, July. The Association for Computer Linguistics.
- Qing Dou and Kevin Knight. 2012. Large scale decipherment for out-of-domain machine translation. In *Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 266–275, Jeju Island, Korea, July. Association for Computational Linguistics.
- Kevin Knight and Kenji Yamada. 1999. A computational approach to deciphering unknown scripts. In *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, number 1, pages 37–44. Association for Computational Linguistics.
- Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proceedings of the Conference on Computational Linguistics and Association of Computation Linguistics (COLING/ACL) Main Conference Poster Sessions*, pages 499–506, Sydney, Australia, July. Association for Computational Linguistics.
- Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2(1-2):83–97.
- Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 156–164, Jeju, Republic of Korea, July. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 812–819, Honolulu, Hawaii. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2011. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 12–21, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Sartaj Sahni and Teofilo Gonzalez. 1976. P-complete approximation problems. *Journal of the Association for Computing Machinery (JACM)*, 23(3):555–565, July.

# Non-Monotonic Sentence Alignment via Semisupervised Learning

Xiaojun Quan, Chunyu Kit and Yan Song

Department of Chinese, Translation and Linguistics

City University of Hong Kong, HKSAR, China

{xiaoquan, ctckit, [yansong]}@[student.]cityu.edu.hk

## Abstract

This paper studies the problem of non-monotonic sentence alignment, motivated by the observation that coupled sentences in real bitexts do not necessarily occur monotonically, and proposes a semisupervised learning approach based on two assumptions: (1) sentences with high affinity in one language tend to have their counterparts with similar relatedness in the other; and (2) initial alignment is readily available with existing alignment techniques. They are incorporated as two constraints into a semisupervised learning framework for optimization to produce a globally optimal solution. The evaluation with real-world legal data from a comprehensive legislation corpus shows that while existing alignment algorithms suffer severely from non-monotonicity, this approach can work effectively on both monotonic and non-monotonic data.

## 1 Introduction

Bilingual sentence alignment is a fundamental task to undertake for the purpose of facilitating many important natural language processing applications such as statistical machine translation (Brown et al., 1993), bilingual lexicography (Klavans et al., 1990), and cross-language information retrieval (Nie et al., 1999). Its objective is to identify correspondences between bilingual sentences in given bitexts. As summarized by Wu (2010), existing sentence alignment techniques rely mainly on sentence length and bilingual lexical resource. Approaches based on the former perform effectively on cognate languages but not on the others. For instance, the statistical correlation of sentence length between English and Chinese is not as high as that between two Indo-European languages (Wu, 1994). Lexicon-based

approaches resort to word correspondences in a bilingual lexicon to match bilingual sentences. A few sentence alignment methods and tools have also been explored to combine the two. Moore (2002) proposes a multi-pass search procedure using both sentence length and an automatically-derived bilingual lexicon. Hunalign (Varga et al., 2005) is another sentence aligner that combines sentence length and a lexicon. Without a lexicon, it backs off to a length-based algorithm and then automatically derives a lexicon from the alignment result. Soon after, Ma (2006) develops the lexicon-based aligner Champollion, assuming that different words have different importance in aligning two sentences.

Nevertheless, most existing approaches to sentence alignment follow the monotonicity assumption that coupled sentences in bitexts appear in a similar sequential order in two languages and crossings are not entertained in general (Langlais et al., 1998; Wu, 2010). Consequently the task of sentence alignment becomes handily solvable by means of such basic techniques as dynamic programming. In many scenarios, however, this prerequisite monotonicity cannot be guaranteed. For example, bilingual clauses in legal bitexts are often coordinated in a way not to keep the same clause order, demanding fully or partially crossing pairings. Figure 1 shows a real excerpt from a legislation corpus. Such monotonicity seriously impairs the existing alignment approaches founded on the monotonicity assumption.

This paper is intended to explore the problem of non-monotonic alignment within the framework of semisupervised learning. Our approach is motivated by the above observation and based on the following two assumptions. First, monolingual sentences with high affinity are likely to have their translations with similar relatedness. Following this assumption, we propose the conception of monolingual consistency which, to the best of

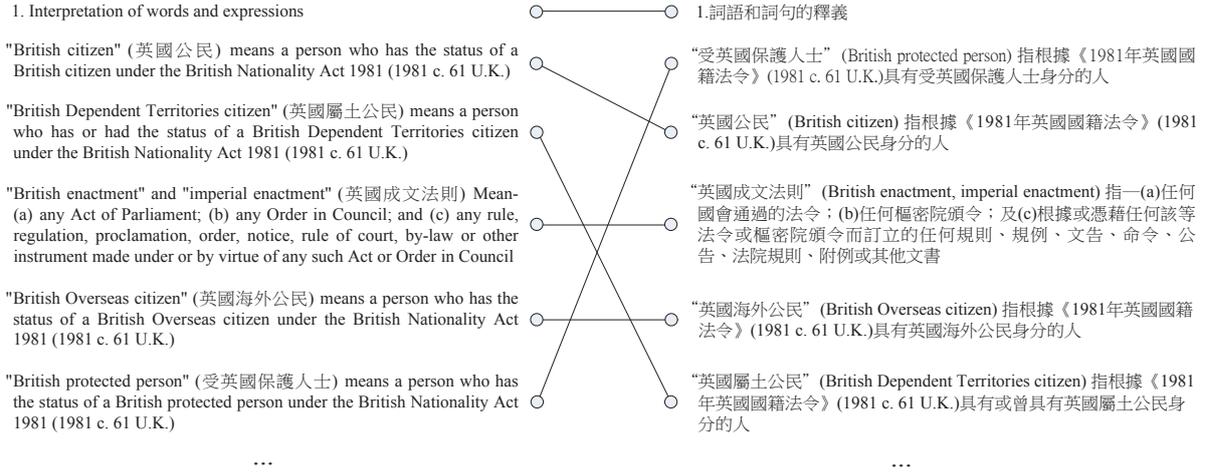


Figure 1: A real example of non-monotonic sentence alignment from BLIS corpus.

our knowledge, has not been taken into account in any previous work of alignment. Second, initial alignment of certain quality can be obtained by means of existing alignment techniques. Our approach attempts to incorporate both monolingual consistency of sentences and bilingual consistency of initial alignment into a semisupervised learning framework to produce an optimal solution. Extensive evaluations are performed using real-world legislation bitexts from BLIS, a comprehensive legislation database maintained by the Department of Justice, HKSAR. Our experimental results show that the proposed method can work effectively while two representatives of existing aligners suffer severely from the non-monotonicity.

## 2 Methodology

### 2.1 The Problem

An alignment algorithm accepts as input a bitext consisting of a set of source-language sentences,  $\mathcal{S} = \{s_1, s_2, \dots, s_m\}$ , and a set of target-language sentences,  $\mathcal{T} = \{t_1, t_2, \dots, t_n\}$ . Different from previous works relying on the monotonicity assumption, our algorithm is generalized to allow the pairings of sentences in  $\mathcal{S}$  and  $\mathcal{T}$  to cross arbitrarily. Figure 2(a) illustrates monotonic alignment with no crossing correspondences in a bipartite graph and 2(b) non-monotonic alignment with scrambled pairings. Note that it is relatively straightforward to identify the type of many-to-many alignment in monotonic alignment using techniques such as dynamic programming if there is no scrambled pairing or the scrambled pairings are local, limited to a short distance. However, the situation of non-monotonic alignment is much

more complicated. Sentences to be merged into a bundle for matching against another bundle in the other language may occur consecutively or discontinuously. For the sake of simplicity, we will not consider non-monotonic alignment with many-to-many pairings but rather assume that each sentence may align to only one or zero sentence in the other language.

Let  $\mathcal{F}$  represent the correspondence relation between  $\mathcal{S}$  and  $\mathcal{T}$ , and therefore  $\mathcal{F} \subset \mathcal{S} \times \mathcal{T}$ . Let matrix  $F$  denote a specific alignment solution of  $\mathcal{F}$ , where  $F_{ij}$  is a real score to measure the likelihood of matching the  $i$ -th sentence  $s_i$  in  $\mathcal{S}$  against the  $j$ -th sentence  $t_j$  in  $\mathcal{T}$ . We then define an alignment function  $\mathcal{A} : F \rightarrow A$  to produce the final alignment, where  $A$  is the alignment matrix for  $\mathcal{S}$  and  $\mathcal{T}$ , with  $A_{ij} = 1$  for a correspondence between  $s_i$  and  $t_j$  and  $A_{ij} = 0$  otherwise.

### 2.2 Semisupervised Learning

A semisupervised learning framework is introduced to incorporate the monolingual and bilingual consistency into alignment scoring

$$Q(F) = Q_m(F) + \lambda Q_b(F), \quad (1)$$

where  $Q_m(F)$  is the term for monolingual constraint to control the consistency of sentences with high affinities,  $Q_b(F)$  for the constraint of initial alignment obtained with existing techniques, and  $\lambda$  is the weight between them. Then, the optimal alignment solution is to be derived by minimizing the cost function  $Q(F)$ , i.e.,

$$F^* = \arg \min_F Q(F). \quad (2)$$

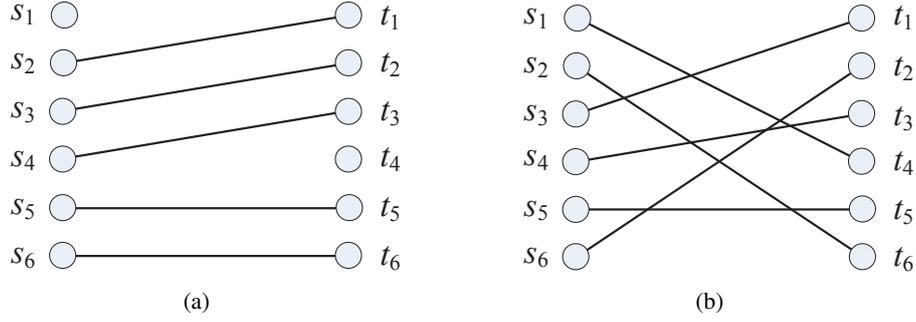


Figure 2: Illustration of monotonic (a) and non-monotonic alignment (b), with a line representing the correspondence of two bilingual sentences.

In this paper,  $\mathcal{Q}_m(F)$  is defined as

$$\frac{1}{4} \sum_{i,j=1}^m W_{ij} \sum_{k,l=1}^n V_{kl} \left( \frac{F_{ik}}{\sqrt{D_{ii}E_{kk}}} - \frac{F_{jl}}{\sqrt{D_{jj}E_{ll}}} \right)^2, \quad (3)$$

where  $W$  and  $V$  are the symmetric matrices to represent the monolingual sentence affinity matrices in  $\mathcal{S}$  and  $\mathcal{T}$ , respectively, and  $D$  and  $E$  are the diagonal matrices with entries  $D_{ii} = \sum_j W_{ij}$  and  $E_{ii} = \sum_j V_{ij}$ . The idea behind (3) is that to minimize the cost function, the translations of those monolingual sentences with close relatedness reflected in  $W$  and  $V$  should also keep similar closeness. The bilingual constraint term  $\mathcal{Q}_b(F)$  is defined as

$$\mathcal{Q}_b(F) = \sum_{i=1}^m \sum_{j=1}^n (F_{ij} - \hat{A}_{ij})^2, \quad (4)$$

where  $\hat{A}$  is the initial alignment matrix obtained by  $\mathcal{A} : \hat{F} \rightarrow \hat{A}$ . Note that  $\hat{F}$  is the initial relation matrix between  $\mathcal{S}$  and  $\mathcal{T}$ .

The monolingual constraint term  $\mathcal{Q}_m(F)$  defined above corresponds to the *smoothness constraint* in the previous semisupervised learning work by Zhou et al. (2004) that assigns higher likelihood to objects with larger similarity to share the same label. On the other hand,  $\mathcal{Q}_b(F)$  corresponds to their *fitting constraint*, which requires the final alignment to maintain the maximum consistency with the initial alignment.

Taking the derivative of  $\mathcal{Q}(F)$  with respect to  $F$ , we have

$$\frac{\partial \mathcal{Q}(F)}{\partial F} = 2F - 2SFT + 2\lambda F - 2\lambda \hat{A}, \quad (5)$$

where  $S$  and  $T$  are the normalized matrices of  $W$  and  $V$ , calculated by  $S = D^{-1/2}WD^{-1/2}$  and

$T = E^{-1/2}VE^{-1/2}$ . Then, the optimal  $F^*$  is to be found by solving the equation

$$(1 + \lambda) F^* - SF^*T = \lambda \hat{A}, \quad (6)$$

which is equivalent to  $\alpha F^* - F^* \beta = \gamma$  with  $\alpha = (1 + \lambda) S^{-1}$ ,  $\beta = T$  and  $\gamma = \lambda S^{-1} \hat{A}$ . This is in fact a Sylvester equation (Barlow et al., 1992), whose numerical solution can be found by many classical algorithms. In this research, it is solved using LAPACK,<sup>1</sup> a software library for numerical linear algebra. Non-positive entries in  $F^*$  indicate unrealistic correspondences of sentences and are thus set to zero before applying the alignment function.

### 2.3 Alignment Function

Once the optimal  $F^*$  is acquired, the remaining task is to design an alignment function  $\mathcal{A}$  to convert it into an alignment solution. An intuitive approach is to use a heuristic search for local optimization (Kit et al., 2004), which produces an alignment with respect to the largest scores in each row and each column. However, this does not guarantee a globally optimal solution. Figure 3 illustrates a mapping relation matrix onto an alignment matrix, which also shows that the optimal alignment cannot be achieved by heuristic search.

Banding is another approach frequently used to convert a relation matrix to alignment (Kay and Röscheisen, 1993). It is founded on the observation that true monotonic alignment paths usually lie close to the diagonal of a relation matrix. However, it is not applicable to our task due to the non-monotonicity involved. We opt for converting a relation matrix into specific alignment by solving

<sup>1</sup><http://www.netlib.org/lapack/>

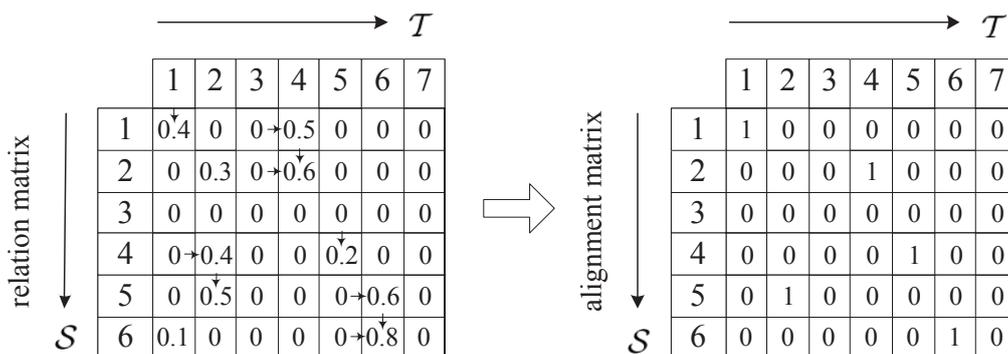


Figure 3: Illustration of sentence alignment from relation matrix to alignment matrix. The scores marked with arrows are the best in each row/column to be used by the heuristic search. The right matrix represents the corresponding alignment matrix by our algorithm.

the following optimization

$$A = \arg \max_X \sum_{i=1}^m \sum_{j=1}^n X_{ij} F_{ij} \quad (7)$$

$$s.t. \sum_{i=1}^m X_{ij} \leq 1, \sum_{j=1}^n X_{ij} \leq 1, X_{ij} \in \{0, 1\}$$

This turns sentence alignment into a problem to be resolved by binary linear programming (BIP), which has been successfully applied to word alignment (Taskar et al., 2005). Given a scoring matrix, it guarantees an optimal solution.

## 2.4 Alignment Initialization

Once the above alignment function is available, the initial alignment matrix  $\hat{A}$  can be derived from an initial relation matrix  $\hat{F}$  obtained by an available alignment method. This work resorts to another approach to initializing the relation matrix. In many genres of bitexts, such as government transcripts or legal documents, there are a certain number of common strings on the two sides of bitexts. In legal documents, for example, translations of many key terms are usually accompanied with their source terms. Also, common numberings can be found in enumerated lists in bitexts. These kinds of *anchor strings* provide quite reliable information to link bilingual sentences into pairs, and thus can serve as useful cues for sentence alignment. In fact, they can be treated as a special type of highly reliable “bilexicon”.

The anchor strings used in this work are derived by searching the bitexts using word-level inverted indexing, a basic technique widely used in information retrieval (Baeza-Yates and Ribeiro-Neto, 2011). For each index term, a list of postings is

created. Each posting includes a sentence identifier, the in-sentence frequency and positions of this term. The positions of terms are intersected to find common anchor strings. The anchor strings, once found, are used to calculate the initial affinity  $\hat{F}_{ij}$  of two sentences using Dice’s coefficient

$$\hat{F}_{ij} = \frac{2|C_{1i} \cap C_{2j}|}{|C_{1i}| + |C_{2j}|} \quad (8)$$

where  $C_{1i}$  and  $C_{2j}$  are the anchor sets in  $s_i$  and  $t_j$ , respectively, and  $|\cdot|$  is the cardinality of a set.

Apart from using anchor strings, other avenues for the initialization are studied in the evaluation section below, i.e., using another aligner and an existing lexicon.

## 2.5 Monolingual Affinity

Although various kinds of information from a monolingual corpus have been exploited to boost statistical machine translation models (Liu et al., 2010; Su et al., 2012), we have not yet been exposed to any attempt to leverage monolingual sentence affinity for sentence alignment. In our framework, an attempt to this can be made through the computation of  $W$  and  $V$ . Let us take  $W$  as an example, where the entry  $W_{ij}$  represents the affinity of sentence  $s_i$  and sentence  $s_j$ , and it is set to 0 for  $i = j$  in order to avoid self-reinforcement during optimization (Zhou et al., 2004).

When two sentences in  $\mathcal{S}$  or  $\mathcal{T}$  are not too short, or their content is not divergent in meaning, their semantic similarity can be estimated in terms of common words. Motivated by this, we define  $W_{ij}$  (for  $i \neq j$ ) based on the Gaussian kernel as

$$W_{ij} = \exp \left( -\frac{1}{2\sigma^2} \left( 1 - \frac{v_i^T v_j}{\|v_i\| \|v_j\|} \right)^2 \right) \quad (9)$$

where  $\sigma$  is the standard deviation parameter,  $v_i$  and  $v_j$  are vectors of  $s_i$  and  $s_j$  with each component corresponding to the *tf-idf* value of a particular term in  $\mathcal{S}$  (or  $\mathcal{T}$ ), and  $\|\cdot\|$  is the norm of a vector. The underlying assumption here is that words appearing frequently in a small number of sentences but rarely in the others are more significant in measuring sentence affinity.

Although semantic similarity estimation is a straightforward approach to deriving the two affinity matrices, other approaches are also feasible. An alternative approach can be based on sentence length under the assumption that two sentences with close lengths in one language tend to have their translations also with close lengths.

## 2.6 Discussion

The proposed semisupervised framework for non-monotonic alignment is in fact generalized beyond, and can also be applied to, monotonic alignment. Towards this, we need to make use of sentence sequence information. One way to do it is to incorporate sentence positions into Equation (1) by introducing a position constraint  $Q_p(F)$  to enforce that bilingual sentences in closer positions should have a higher chance to match one another. For example, the new constraint can be defined as

$$Q_p(F) = \sum_{i=1}^m \sum_{j=1}^n |p_i - q_j| F_{ij}^2,$$

where  $p_i$  and  $q_j$  are the absolute (or relative) positions of two bilingual sentences in their respective sequences. Another way follows the banding assumption that the actual couplings only appear in a narrow band along the main diagonal of relation matrix. Accordingly, all entries of  $F^*$  outside this band are set to zero before the alignment function is applied. Kay and Röscheisen (1993) illustrate that this can be done by modeling the maximum deviation of true couplings from the diagonal as  $O(\sqrt{n})$ .

## 3 Evaluation

### 3.1 Data Set

Our data set is acquired from the Bilingual Laws Information System (BLIS),<sup>2</sup> an electronic database of Hong Kong legislation maintained by the Department of Justice, HKSAR. BLIS

<sup>2</sup><http://www.legislation.gov.hk>

provides Chinese-English bilingual texts of ordinances and subsidiary legislation in effect on or after 30 June 1997. It organizes the legal texts into a hierarchy of chapters, sections, subsections, paragraphs and subparagraphs, and displays the content of a such hierarchical construct (usually a section) on a single web page.

By web crawling, we have collected in total 31,516 English and 31,405 Chinese web pages, forming a bilingual corpus of 31,401 bitexts after filtering out null pages. A text contains several to two hundred sentences. Many bitexts exhibit partially non-monotonic order of sentences. Among them, 175 bitexts are randomly selected for manual alignment. Sentences are identified based on punctuations. OpenNLP Tokenizer<sup>3</sup> is applied to segment English sentences into tokens. For Chinese, since there is no reliable segmenter for this genre of text, we have to treat each Chinese character as a single token. In addition, to calculate the monolingual sentence affinity, stemming of English words is performed with the Porter Stemmer (Porter, 1980) after anchor string mining.

The manual alignment of the evaluation data set is performed upon the initial alignment by Hunalign (Varga et al., 2005), an effective sentence aligner that uses both sentence length and a bilexicon (if available). For this work, Hunalign relies solely on sentence length. Its output is then double-checked and corrected by two experts in bilingual studies, resulting in a data set of 1747 1-1 and 70 1-0 or 0-1 sentence pairs.

The standard deviation  $\sigma$  in (9) is an important parameter for the Gaussian kernel that has to be determined empirically (Zhu et al., 2003; Zhou et al., 2004). In addition, the  $Q$  function also involves another parameter  $\lambda$  to adjust the weight of the bilingual constraint. This work seeks an approach to deriving the optimal parameters without any external training data beyond the initial alignment. A three-fold cross-validation is thus performed on the initial 1-1 alignment and the parameters that give the best average performance are chosen.

### 3.2 Monolingual Consistency

To demonstrate the validity of the monolingual consistency, the semantic similarity defined by  $\frac{v_i^T v_j}{\|v_i\| \|v_j\|}$  is evaluated as follows. 500 pairs of English sentences with the highest similarities are selected, excluding *null* pairings (1-0 or 0-1 type).

<sup>3</sup><http://opennlp.apache.org/>

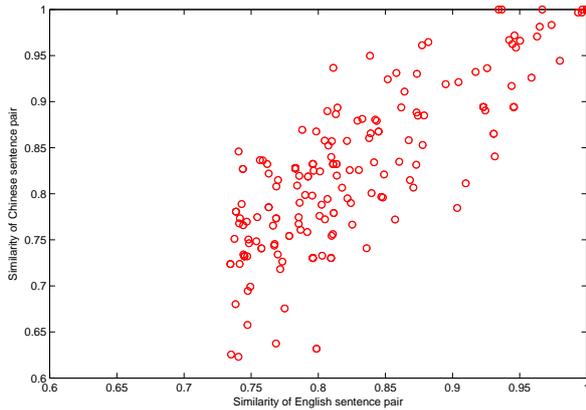


Figure 4: Demonstration of monolingual consistency. The horizontal axis is the similarity of English sentence pairs and the vertical is the similarity of the corresponding pairs in Chinese.

Type	Total	initAlign		NonmoAlign	
		Pred	Corr	Pred	Corr
1-0	70	662	66	70	50
1-1	1747	1451	1354	1747	1533

Table 1: Performance of the initial alignment and our aligner, where the *Pred* and *Corr* columns are the numbers of predicted and correct pairings.

All of these high-affinity pairs have a similarity score higher than 0.72. A number of duplicate sentences (e.g., date) with exceptionally high similarity 1.0 are dropped. Also, the similarity of the corresponding translations of each selected pair is calculated. These two sets of similarity scores are then plotted in a scatter plot, as in Figure 4. If the monolingual consistency assumption holds, the plotted points would appear nearby the diagonal. Figure 4 confirms this, indicating that sentence pairs with high affinity in one language do have their counterparts with similarly high affinity in the other language.

### 3.3 Impact of Initial Alignment

The 1-1 initial alignment plays the role of labeled instances for the semisupervised learning. It is of critical importance to the learning performance. As shown in Table 1, our alignment function predicts 1451 1-1 pairings by virtue of anchor strings, among which 1354 pairings are correct, yielding a relatively high precision in the non-monotonic circumstance. It also predicts *null* alignment for many sentences that contain no anchor. This explains why it outputs 662 1-0 pairings when there

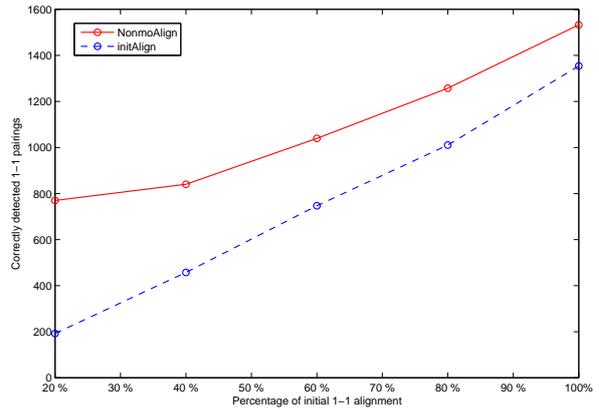


Figure 5: Performance of non-monotonic alignment along the percentage of initial 1-1 alignment.

are only 70 1-0 true ones. Starting from this initial alignment, our aligner (let us call it *NonmoAlign*) discovers 179 more 1-1 pairings.

A question here is concerned with how the scale of initial alignment affects the final alignment. To examine this, we randomly select 20%, 40%, 60% and 80% of the 1451 1-1 detected pairings as the initial alignments for a series of experiments. The random selection for each proportion is performed ten times and their average alignment performance is taken as the final result and plotted in Figure 5. An observation from this figure is that the aligner consistently discovers significantly more 1-1 pairings on top of an initial 1-1 alignment, which has to be accounted for by the monolingual consistency. Another observation is that the alignment performance goes up along the increase of the percentage of initial alignment while performance gain slows down gradually. When the percentage is very low, the aligner still works quite effectively.

### 3.4 Non-Monotonic Alignment

To test our aligner with non-monotonic sequences of sentences, we have them randomly scrambled in our experimental data. This undoubtedly increases the difficulty of sentence alignment, especially for the traditional approaches critically relying on monotonicity.

The baseline methods used for comparison are Moore’s aligner (Moore, 2002) and Hunalign (Varga et al., 2005). Hunalign is configured with the option [-realign], which triggers a three-step procedure: after an initial alignment, Hunalign heuristically enriches its dictionary using word co-occurrences in identified sentence pairs; then, it re-runs the alignment process using the updated

Type	Moore			Hunalign			NonmoAlign		
	$P$	$R$	$F_1$	$P$	$R$	$F_1$	$P$	$R$	$F_1$
1-1	0.104	0.104	0.104	0.407	0.229	0.293	0.878	0.878	0.878
1-0	0.288	0.243	0.264	0.033	0.671	0.062	0.714	0.714	0.714
Micro	0.110	0.110	0.110	0.184	0.246	0.210	0.871	0.871	0.871

Table 2: Performance comparison with the baseline methods.

dictionary. According to Varga et al (2005), this setting gives a higher alignment quality than otherwise. In addition, Hunalign can use an external bilexicon. For a fair comparison, the identified anchor set is fed to Hunalign as a special bilexicon. The performance of alignment is measured by precision ( $P$ ), recall ( $R$ ) and F-measure ( $F_1$ ). Micro-averaged performance scores of precision, recall and F-measure are also computed to measure the overall performance on 1-1 and 1-0 alignment. The final results are presented in Table 2, showing that both Moore’s aligner and Hunalign underperform ours on non-monotonic alignment. The particularly poor performance of Moore’s aligner has to be accounted for by its requirement of more than thousands of sentences in bitext input for reliable estimation of its parameters. Unfortunately, our available data has not reached that scale yet.

### 3.5 Partially Non-Monotonic Alignment

Full non-monotonic bitexts are rare in practice. But partial non-monotonic ones are not. Unlike traditional alignment approaches, ours does not found its performance on the degree of monotonicity. To test this, we construct five new versions of the data set for a series of experiments by randomly choosing and scrambling 0%, 10%, 20%, 40%, 60% and 80% sentence parings. In theory, partial non-monotonicity of various degrees should have no impact on the performance of our aligner. It is thus not surprised that it achieves the same result as reported in last subsection. NonmoAlign initialized with Hunalign (marked as NonmoAlign\_Hun) is also tested. The experimental results are presented in Figure 6. It shows that both Moore’s aligner and Hunalign work relatively well on bitexts with a low degree of non-monotonicity, but their performance drops dramatically when the non-monotonicity is increased. Despite the improvement at low non-monotonicity by seeding our aligner with Hunalign, its performance decreases likewise when the degree of non-monotonicity increases, due to the quality de-

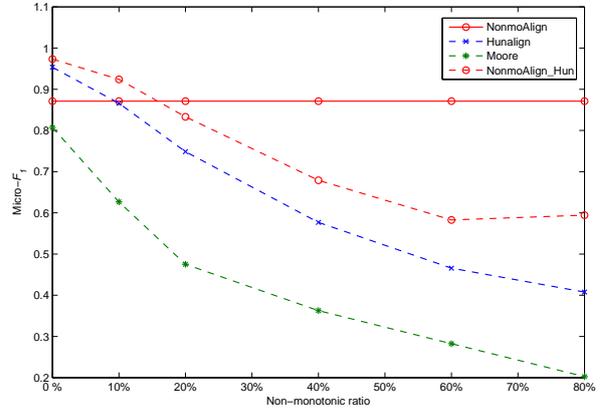


Figure 6: Performance of alignment approaches at different degrees of non-monotonicity.

crease of the initial alignment by Hunalign.

### 3.6 Monotonic Alignment

The proposed alignment approach is also expected to work well on monotonic sentence alignment. An evaluation is conducted for this using a monotonic data set constructed from our data set by discarding all its 126 crossed pairings. Of the two strategies discussed above, banding is used to help our aligner incorporate the sequence information. The initial relation matrix is built with the aid of a dictionary automatically derived by Hunalign. Entries of the matrix are derived by employing a similar strategy as in Varga et al. (2005). The evaluation results are presented in Table 3, which shows that NonmoAlign still achieves very competitive performance on monotonic sentence alignment.

## 4 Related Work

The research of sentence alignment originates in the early 1990s. Gale and Church (1991) and Brown (1991) report the early works using length statistics of bilingual sentences. The general idea is that the closer two sentences are in length, the more likely they are to align. A notable difference of their methods is that the former uses sentence

Type	Moore			Hunalign			NonmoAlign		
	$P$	$R$	$F_1$	$P$	$R$	$F_1$	$P$	$R$	$F_1$
1-1	0.827	0.828	0.827	0.999	0.972	0.986	0.987	0.987	0.987
1-0	0.359	0.329	0.343	0.330	0.457	0.383	0.729	0.729	0.729
Micro	0.809	0.807	0.808	0.961	0.951	0.956	0.976	0.976	0.976

Table 3: Performance of monotonic alignment in comparison with the baseline methods.

length in number of characters while the latter in number of tokens. Both use dynamic programming to search for the best alignment. As shown in Chen (1993) and Wu (1994), however, sentence-length based methods suffer when the texts to be aligned contain small passages, or the languages involved share few cognates. The subsequent stage of sentence alignment research is accompanied by the advent of a handful of well-designed alignment tools. Moore (2002) proposes a three-pass procedure to find final alignment. Its bitext input is initially aligned based on sentence length. This step generates a set of strictly-selected sentence pairs for use to train an IBM translation model 1 (Brown et al., 1993). Its final step realigns the bitext using both sentence length and the discovered word correspondences. Hunalign (Varga et al., 2005), originally proposed as an ingredient for building parallel corpora, has demonstrated an outstanding performance on sentence alignment. Like many other aligners, it employs a similar strategy of combining sentence length and lexical data. In the absence of a lexicon, it first performs an initial alignment wholly relying on sentence length and then automatically builds a lexicon based on this alignment. Using an available lexicon, it produces a rough translation of the source text by converting each token to the one of its possible counterparts that has the highest frequency in the target corpus. Then, the relation matrix of a bitext is built of similarity scores for the rough translation and the actual translation at sentence level. The similarity of two sentences is calculated in terms of their common pairs and length ratio.

To deal with noisy input, Ma (2006) proposes a lexicon-based sentence aligner - Champollion. Its distinctive feature is that it assigns different weights to words in terms of their *tf-idf* scores, assuming that words with low sentence frequencies in a text but high occurrences in some local sentences are more indicative of alignment. Under this assumption, the similarity of any two sentences is calculated accordingly and then a dy-

namic programming algorithm is applied to produce final alignment. Following this work, Li et al. (2010) propose a revised version of Champollion, attempting to improve its speed without performance loss. For this purpose, the input bitexts are first divided into smaller aligned fragments before applying Champollion to derive finer-grained sentence pairs. In another related work by Deng et al. (2007), a generative model is proposed, accompanied by two specific alignment strategies, i.e., dynamic programming and divisive clustering. Although a non-monotonic search process that tolerates two successive chunks in reverse order is involved, their work is essentially targeted at monotonic alignment.

## 5 Conclusion

In this paper we have proposed and tested a semisupervised learning approach to non-monotonic sentence alignment by incorporating both monolingual and bilingual consistency. The utility of monolingual consistency in maintaining the consonance of high-affinity monolingual sentences with their translations has been demonstrated. This work also exhibits that bilingual consistency of initial alignment of certain quality is useful to boost alignment performance. Our evaluation using real-world data from a legislation corpus shows that the proposed approach outperforms the baseline methods significantly when the bitext input is composed of non-monotonic sentences. Working on partially non-monotonic data, this approach also demonstrates a superior performance. Although initially proposed for non-monotonic alignment, it works well on monotonic alignment by incorporating the constraint of sentence sequence.

## Acknowledgments

The research described in this paper was substantially supported by the Research Grants Council (RGC) of Hong Kong SAR, China, through the GRF grant 9041597 (CityU 144410).

## References

- Ricardo Baeza-Yates and Berthier Ribeiro-Neto. 2011. *Modern Information Retrieval: The Concepts and Technology Behind Search*, 2nd ed., Harlow: Addison-Wesley.
- Jewel B. Barlow, Moghen M. Monahemi, and Dianne P. O’Leary. 1992. Constrained matrix Sylvester equations. In *SIAM Journal on Matrix Analysis and Applications*, 13(1):1-9.
- Peter F. Brown, Jennifer C. Lai, Robert L. Mercer. 1991. Aligning sentences in parallel corpora. In *Proceedings of ACL’91*, pages 169-176.
- Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263-311.
- Stanley F. Chen. 1993. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of ACL’93*, pages 9-16.
- Yonggang Deng, Shankar Kumar, and William Byrne. 2007. Segmentation and alignment of parallel text for statistical machine translation. *Natural Language Engineering*, 13(3): 235-260.
- William A. Gale, Kenneth Ward Church. 1991. A Program for aligning sentences in bilingual corpora. In *Proceedings of ACL’91*, pages 177-184.
- Martin Kay and Martin Röscheisen. 1993. Text-translation alignment. *Computational Linguistics*, 19(1):121-142.
- Chunyu Kit, Jonathan J. Webster, King Kui Sin, Haihua Pan, and Heng Li. 2004. Clause alignment for bilingual HK legal texts: A lexical-based approach. *International Journal of Corpus Linguistics*, 9(1):29-51.
- Chunyu Kit, Xiaoyue Liu, King Kui Sin, and Jonathan J. Webster. 2005. Harvesting the bitexts of the laws of Hong Kong from the Web. In *The 5th Workshop on Asian Language Resources*, pages 71-78.
- Judith L. Klavans and Evelyne Tzoukermann. 1990. The bicord system: Combining lexical information from bilingual corpora and machine readable dictionaries. In *Proceedings of COLING’90*, pages 174-179.
- Philippe Langlais, Michel Simard, and Jean Véronis. 1998. Methods and practical issues in evaluating alignment techniques. In *Proceedings of COLING-ACL’98*, pages 711-717.
- Zhanyi Liu, Haifeng Wang, Hua Wu, and Sheng Li. 2010. Improving statistical machine translation with monolingual collocation. In *Proceedings of ACL 2010*, pages 825-833.
- Xiaoyi Ma. 2006. Champollion: A robust parallel text sentence aligner. In *LREC 2006*, pages 489-492.
- Peng Li, Maosong Sun, Ping Xue. 2010. Fast-Champollion: a fast and robust sentence alignment algorithm. In *Proceedings of ACL 2010: Posters*, pages 710-718.
- Robert C. Moore. 2002. Fast and accurate sentence alignment of bilingual corpora. In *Proceedings of AMTA 2002*, page 135-144.
- Jian-Yun Nie, Michel Simard, Pierre Isabelle and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the Web. In *Proceedings of SIGIR’99*, pages 74-81.
- Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3): 130-137.
- Jinsong Su, Hua Wu, Haifeng Wang, Yidong Chen, Xiaodong Shi, Huailin Dong, Qun Liu. 2012. Translation model adaptation for statistical machine translation with monolingual topic information. In *Proceedings of ACL 2012*, Vol. 1, pages 459-468.
- Ben Taskar, Simon Lacoste-Julien and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proceedings of HLT/EMNLP 2005*, pages 73-80.
- Dániel Varga, Péter Halácsy, András Kornai, Viktor Nagy, László Németh, Viktor Trón. 2005. Parallel corpora for medium density languages. In *Proceedings of RANLP 2005*, pages 590-596.
- Dekai Wu. 1994. Aligning a parallel English-Chinese corpus statistically with lexical criteria. In *Proceedings of ACL’94*, pages 80-87.
- Dekai Wu. 2010. Alignment. *Handbook of Natural Language Processing*, 2nd ed., CRC Press.
- Dengyong Zhou, Olivier Bousquet, Thomas N. Lal, Jason Weston, Bernhard Schölkopf. 2004. Learning with local and global consistency. *Advances in Neural Information Processing Systems*, 16:321-328.
- Xiaojin Zhu, Zoubin Ghahramani and John Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of ICML 2003*, pages 912-919.

# Bootstrapping Entity Translation on Weakly Comparable Corpora

Taesung Lee and Seung-won Hwang

Department of Computer Science and Engineering  
Pohang University of Science and Technology (POSTECH)  
Pohang, Republic of Korea  
{elca4u, swhwang}@postech.edu

## Abstract

This paper studies the problem of mining named entity translations from comparable corpora with some “asymmetry”. Unlike the previous approaches relying on the “symmetry” found in parallel corpora, the proposed method is tolerant to asymmetry often found in comparable corpora, by distinguishing different semantics of relations of entity pairs to selectively propagate seed entity translations on weakly comparable corpora. Our experimental results on English-Chinese corpora show that our selective propagation approach outperforms the previous approaches in named entity translation in terms of the mean reciprocal rank by up to 0.16 for organization names, and 0.14 in a low comparability case.

## 1 Introduction

Identifying and understanding entities is a crucial step in understanding text. This task is more challenging in the presence of multilingual text, because translating named entities (NEs), such as persons, locations, or organizations, is a non-trivial task. Early research on NE translation used phonetic similarities, for example, to mine the translation ‘Mandelson’ → ‘曼德尔森’ [ManDeErSen] with similar sounds. However, not all NE translations are based on transliterations, as shown in Table 1—Some translations, especially the names of most organizations, are based on semantic equivalences. Furthermore, names can be abbreviated in one or both languages, e.g., the ‘World Trade Organization’ (世界贸易组织) can be called the ‘WTO’ (世贸组织). Another challenging example is that, a translation can be arbitrary, e.g., ‘Jackie Chan’ → ‘成龙’ [ChengLong]. There are many approaches

English	Chinese
World Trade Organization	世界贸易组织
WTO	[ShiJieMaoYiZuZhi] 世贸组织 [ShiMaoZuZhi]
Jackie Chan	成龙 [ChengLong]

Table 1: Examples of non-phonetic translations.

that deal with some of these challenges (Lam et al., 2007; Yang et al., 2009), e.g., by combining phonetic similarity and a dictionary. However, arbitrary translations still cannot be handled by examining the NE pair itself. Corpus-based approaches (Kupiec, 1993; Feng, 2004), by mining external signals from a large corpus, such as parathetical translation “成龙 (Jackie Chan)”, complement the problem of transliteration-based approaches, but the coverage of this approach is limited to popular entities with such evidence.

The most effective known approach to NE translation has been a holistic framework (You et al., 2010; Kim et al., 2011; You et al., 2012) combining transliteration- and corpus-based methods. In these approaches, both 1) arbitrary translations and 2) lesser-known entities can be handled, by propagating the translation scores of known entities to lesser-known entities if they co-occur frequently in both corpora. For example, a lesser-known entity Tom Watson can be translated if Mandelson and Tom Watson co-occur frequently in an English corpus, and their Chinese translations also co-occur frequently in a Chinese corpus, i.e., if the co-occurrences in the two corpora are “symmetric”.

A research question we ask in this paper is: What if comparable corpora are not comparable enough to support this *symmetry* assumption? We found that this is indeed the case. For example, even English and Chinese news from the same publisher may have different focus—the Chinese version focuses more on Chinese Olympic

teams and Chinese local news. In the presence of such asymmetry, all previous approaches, building upon symmetry, quickly deteriorate by propagating false positives. For example, co-occurrence of Mandelson and Tom Watson may not appear in a Chinese corpus, which may lead to the translation of Tom Watson into another Chinese entity Gordon Brown which happens to co-occur with the Chinese translation of Mandelson.

Our key contribution is to avoid such false propagation, by discerning the semantics of relations. For example, relations between Mandelson and Tom Watson, should be semantically different from Chinese relations between ‘戈登·布朗’ (Gordon Brown) and ‘曼德尔森’ (Mandelson). A naive approach would be finding documents with a similar *topic* such as politics, and scientific discovery, and allowing propagation only when the topic agrees. However, we found that a topic is a unit that is too coarse for this task because most articles on Mandelson will invariably fall into the same topic<sup>1</sup>. In clear contrast, we *selectively propagate* seed translations, only when the relations in the two corpora share the same semantics.

This selective propagation can be especially effective for translating challenging types of entities such as *organizations* including the WTO used with and without abbreviation in both languages. Applying a holistic approach (You et al., 2012) on organizations leads to poor results, 0.06 in terms of the F1-score. A naive approach to increase the precision would be to consider multi-type co-occurrences, hoping that highly precise translations of some type, e.g., persons with an F1-score of 0.69 (You et al., 2012), can be propagated to boost the precision on organizations. In our experiments, this naive multi-type propagation still leads to an unsatisfactory F1-score of 0.12. Such a low score can be explained by the following example. When translating ‘WTO’ using the co-occurrence with ‘Mandelson’, other co-occurrences such as (London, Mandelson) and (EU, Mandelson) produce a lot of noise because the right translation of WTO does not share much phonetic/semantic similarity. Our understanding of relation semantics, can distinguish “Mandelson was born in London” from “Mandelson visited the WTO”, to stop false propagations, which generates an F1-score 0.25 higher than the existing ap-

<sup>1</sup>The MRR for organization names achieved by a topic model-based approach was 0.15 lower than our best.

proaches.

More formally, we enable *selective propagation* of seed translations on weakly comparable corpora, by 1) clarifying the detailed meaning of relational information of co-occurring entities, and 2) identifying the contexts of the relational information using statement-level context comparison. In other words, we propagate the translation score of a known translation pair to a neighbor pair *if* the semantics of their relations in English and Chinese corpora are *equivalent* to accurately propagate the scores. For example, if we know ‘Russia’ → ‘俄罗斯’<sub>(1)</sub> and *join* → 加入<sub>(2)</sub>, then from a pair of statements “Russia<sub>(1)</sub> joins<sub>(2)</sub> the WTO<sub>(3)</sub>” and “俄罗斯<sub>(1)</sub> 加入<sub>(2)</sub> 世贸组织<sub>(3)</sub>”, we can propagate the translation score of (Russia, 俄罗斯)<sub>(1)</sub> to (WTO, 世贸组织)<sub>(3)</sub>. However, we do not exploit a pair of statements “Russia joined the WTO” and “俄罗斯谴责<sub>(2)</sub> 摩洛哥” because 谴责<sub>(2)</sub> does not mean *join*<sub>(2)</sub>. Furthermore, we mine a similar English-Chinese document pair that can be found by comparing the entity relationships, such as “Mandelson visited Moscow” and “Mandelson met Alexei Kudrin”, within the English document and the Chinese document to leverage similar contexts to assure that we use symmetric parts.

For this goal, we first extract *relations* among entities in documents, such as *visit* and *join*, and mine semantically equivalent relations across the languages, e.g., English and Chinese, such as *join* → 加入. Once these relation translations are mined, similar document pairs can be identified by comparing each constituent relationship among entities using their relations. Knowing document similarity improves NE translation, and improved NE translation can boost the accuracy of document and relationship similarity. This iterative process can continue until convergence.

To the best of our knowledge, our approach is the first to translate a broad range of multilingual relations and exploit them to enhance NE translation. In particular, our approach leverages semantically similar document pairs to exclude incomparable parts that appear in one language only. Our method outperforms the previous approaches in translating NE up to 0.16 in terms of the mean reciprocal rank (MRR) for organization names. Moreover, our method shows robustness, with 0.14 higher MRR than seed translations, on less comparable corpora.

## 2 Related Work

This work is related to two research streams: NE translation and semantically equivalent relation mining.

### Entity translation

Existing approaches on NE translation can be categorized into 1) transliteration-based, 2) corpus-based, and 3) hybrid approaches.

Transliteration-based approaches (Wan and Verspoor, 1998; Knight and Graehl, 1998) are the foundations of many decent methods, but they alone suffer from ambiguity (e.g., 史蒂夫 and 始第夫 have the same sounds) and cannot handle non-transliterated cases such as ‘Jackie Chan (成龙[ChengLong])’. Some methods (Lam et al., 2007; Yang et al., 2009) rely on meanings of constituent letters or words to handle organization name translation such as ‘Bank of China (中国银行)’, whose translation is derived from ‘China (中国)’, and ‘a bank (银行)’. However, many names often originate from abbreviation (such as ‘WTO’); hence we cannot always leverage meanings.

Corpus-based approaches (Kupiec, 1993; Lin et al., 2008; Jiang et al., 2009) exploit high-quality bilingual evidence such as parenthetical translation, e.g., “成龙 (Jackie Chan)”, (Lin et al., 2008), semi-structural patterns (Jiang et al., 2009), and parallel corpus (Kupiec, 1993). However, the coverage of the corpus-based approaches is limited to popular entities with such bilingual evidences. On the other hand, our method can cover entities with monolingual occurrences in corpora, which significantly improves the coverage.

The most effective known approach is a holistic framework that combines those two approaches (You et al., 2012; You et al., 2010; Kim et al., 2011). You et al. (2010; 2012) leverage two graphs of entities in each language, that are generated from a pair of corpora, with edge weights quantified as the strength of the relatedness of entities. Then, two graphs are iteratively aligned using the common neighbors of two entities. Kim et al. (2011) build such graphs using the context similarity, measured with a bag of words approach, of entities in news corpora to translate NEs. However, these approaches assume the symmetry of the two graphs. This assumption holds if two corpora are parallel, but such resources are scarce. But our approach exploits comparable parts from corpora.

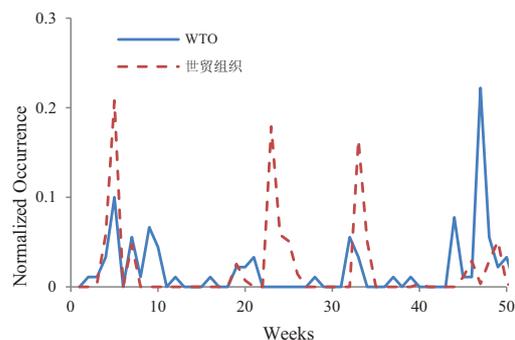


Figure 1: Dissimilarity of temporal distributions of ‘WTO’ in English and Chinese corpora.

Other interesting approaches such as (Klementiev and Roth, 2006; Kim et al., 2012) rely on temporal distributions of entities. That is, two entities are considered to be similar if the two entities in different languages have similar occurrence distributions over time. However, the effectiveness of this feature also depends on the comparability of entity occurrences in time-stamped corpora, which may not hold as shown in Figure 1. In clear contrast, our method can find and compare articles, on different dates, describing the same NE. Moreover, our method does not require time stamps.

### Semantically similar relation mining

Recently, similar relation mining in one language has been studied actively as a key part of automatic knowledge base construction. In automatically constructed knowledge bases, finding semantically similar relations can improve understanding of the Web describing content with many different expressions. As such an effort, PATTY (Nakashole et al., 2012) finds similar relations with almost the same support sets—the sets of NE pairs that co-occur with the relations. However, because of the regional locality of information, bilingual corpora contain many NE pairs that appear in only one of the support sets of the semantically identical relations. NELL (Mohamed et al., 2011) finds related relations using seed pairs of one given relation; then, using K-means clustering, it finds relations that are semantically similar to the given relation. Unfortunately, this method requires that we set K manually, and extract relations for each given relation. Therefore, this is unsuitable to support general relations.

There are only few works on translating relations or obtaining multi-lingual similar relations. Schone et al. (2011) try to find relation patterns

in multiple languages for given seed pairs of a relation. Because this approach finds seed pairs in Wikipedia infoboxes, the number of retrievable relations is restricted to five. Kim et al. (2010) seek more diverse types of relations, but it requires parallel corpora, which are scarce.

### 3 Framework Overview

In this section, we provide an overview of our framework for translating NEs, using news corpora in English and Chinese as a running example. Because such corpora contain asymmetric parts, the goal of our framework is to overcome asymmetry by distinguishing the semantics of relations, and leveraging document context defined by the relations of entities.

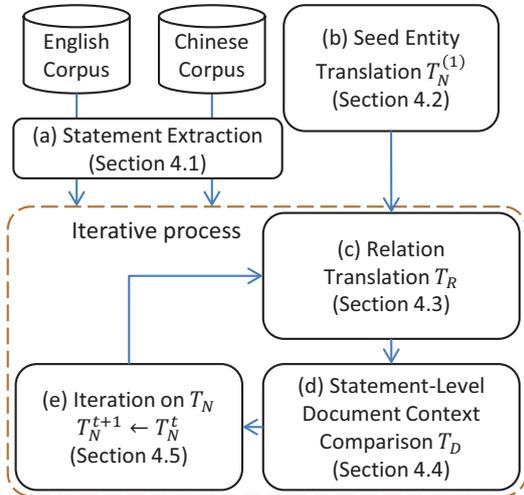


Figure 2: Framework overview.

For this purpose, we build a mutual bootstrapping framework (Figure 2), between entity translation and relation translation using extracted relationships of entities (Figure 2 (a), Section 4.1). More formally, we use the following process:

1. Base condition (Figure 2 (a), Section 4.2): Initializing  $T_N^{(1)}(e_E, e_C)$ , a seed entity translation score, where  $e_E$  is an English entity, and  $e_C$  is a Chinese entity.  $T_N^{(1)}$  can be initialized by phonetic similarity or other NE translation methods.
2. Iteration: Obtaining  $T_N^{t+1}$  using  $T_N^t$ .
  - 1) Using  $T_N^t$ , we obtain a set of relation translations with a semantic similarity score,  $T_R^t(r_E, r_C)$ , for an English relation  $r_E$  and a Chinese relation  $r_C$  (Figure 2 (b), Section 4.3) (e.g.,  $r_E = \text{visit}$  and  $r_C = \text{访问}$ ).

- 2) Using  $T_N^t$  and  $T_R^t$ , we identify a set of semantically similar document pairs that describe the same event with a similarity score  $T_D^t(d_E, d_C)$  where  $d_E$  is an English document and  $d_C$  is a Chinese document (Figure 2 (c), Section 4.4).
- 3) Using  $T_N^t$ ,  $T_R^t$  and  $T_D^t$ , we compute  $T_N^{t+1}$ , an improved entity translation score (Figure 2 (d), Section 4.5).

Each sub-goal reinforces the result of others in the  $(t + 1)$ -th iteration, and by iteratively running them, we can improve the quality of translations. Note that, hereinafter, we omit  $(t)$  for readability when there is no ambiguity.

## 4 Methods

In this section, we describe our method in detail. First, we explain how we extract statements, which are units of relational information, from documents in Section 4.1, and how we obtain seed name translations in Section 4.2. Next, we present our method for discovering relation translations across languages in Section 4.3. In Section 4.4, we use the name translations and the relation translations to compare document contexts which can boost the precision of NE translation. In Section 4.5, we describe how we use the resources obtained so far to improve NE translation.

### 4.1 Statement Extraction

We extract relational statements, which we exploit to propagate translation scores, from an English news corpus and a Chinese news corpus. A *relational statement*, or simply a *statement* is a triple  $(x, r, y)$ , representing a relationship between two names,  $x$  and  $y$ . For example, from “Mandelson recently visited Moscow,” we obtain this statement: (Mandelson, visit, Moscow). We follow a standard procedure to extract statements, as similarly adopted by Nakashole et al. (2012), using Stanford CoreNLP (Klein and Manning, 2003) to lemmatize and parse sentences. Here, we refer readers to existing work for further details because this is not our key contribution.

### 4.2 Seed Entity Translation

We need a few seed translation pairs to initiate the framework. We build a seed translation score  $T_N^{(1)}(e_E, e_C)$  indicating the similarity of an English entity  $e_E$  and a Chinese entity  $e_C$  using an existing method. For example, most methods would give high value for

$T_N^{(1)}$  (Mandelson, 曼德尔森 [ManDeErSen]). In this work, we adopted (You et al., 2012) with (Lam et al., 2007) as a base translation matrix to build the seed translation function. We also use a dictionary to obtain non-NE translations such as ‘government’. We use an English-Chinese general word dictionary containing approximately 80,000 English-Chinese translation word pairs that was also used by Kim et al. (2011) to measure the similarity of context words of entities.

### 4.3 Relation Translation

We need to identify relations that have the equivalent semantics across languages, (e.g., *visit* → 访问), to enable selective propagation of translation scores. Formally, our goal is to measure a pairwise relation translation score  $T_R(r_E, r_C)$  for an English relation  $r_E \in R_E$  and a Chinese relation  $r_C \in R_C$  where  $R_E$  is a set of all English relations and  $R_C$  is a set of all Chinese relations.

We first explain a basic feature to measure the similarity of two relations, its limitations, and how we address the problems. A basic clue is that relations of the same meaning are likely to be mentioned with the same entity pairs. For example, if we have (Mandelson, visit, Moscow) as well as (Mandelson, head to, Moscow) in the corpus, this is a positive signal that the two relations may share the same meaning. Such NE pairs are called *support pairs* of the two relations.

We formally define this clue for relations in the same language, and then describe that in the bilingual setting. A *support intersection*  $H_m(r^i, r^j)$ , a set of support pairs, for monolingual relations  $r^i$  and  $r^j$  is defined as

$$H_m(r^i, r^j) = H(r^i) \cap H(r^j) \quad (1)$$

where  $H(r)$  is the *support set* of a relation  $r$  defined as  $H(r) = \{(x, y) | (x, r, y) \in \mathbf{S}\}$ , and  $\mathbf{S}$  is either  $\mathbf{S}_E$ , a set of all English statements, or  $\mathbf{S}_C$ , a set of all Chinese statements that we extracted in Section 4.1.

Likewise, we can define a support intersection for relations in the different languages using the translation score  $T_N(e_E, e_C)$ . For an English relation  $r_E$  and a Chinese relation  $r_C$ ,

$$H_b(r_E, r_C) = \{(x_E, x_C, y_E, y_C) | \begin{aligned} &T_N(x_E, x_C) \geq \theta \\ &\text{and } T_N(y_E, y_C) \geq \theta \\ &\text{for } (x_E, r_E, y_E) \in \mathbf{S}_E \\ &\text{and } (x_C, r_C, y_C) \in \mathbf{S}_C \end{aligned}\} \quad (2)$$

where  $\theta = 0.6$  is a harsh threshold to exclude most of the false translations by  $T_N$ .

Finally, we define a support intersection, a set of support pairs between two relations  $r^i$  and  $r^j$  of any languages,

$$H(r^i, r^j) = \begin{cases} H_b(r^i, r^j) & \text{if } r^i \in R_E \text{ and } r^j \in R_C \\ H_b(r^j, r^i) & \text{if } r^j \in R_E \text{ and } r^i \in R_C \\ H_m(r^i, r^j) & \text{otherwise} \end{cases} \quad (3)$$

Intuitively,  $|H(r^i, r^j)|$  indicates the strength of the semantic similarity of two relations  $r^i$  and  $r^j$  of any languages. However, as shown in Table 2, we cannot use this value directly to measure the similarity because the support intersection of semantically similar *bilingual* relations (e.g.,  $|H(\text{head to}, \text{访问})| = 2$ ) is generally very low, and normalization cannot remedy this problem as we can see from  $|H(\text{visit}, \text{访问})| = 27$  and  $|H(\text{visit})| = 1617$ .

Set	Cardinality
$H(\text{visit})$	1617
$H(\text{访问})$	2788
$H(\text{visit}, \text{访问})$	27
$H(\text{head to}, \text{访问})$	2

Table 2: Evidence cardinality in the corpora.

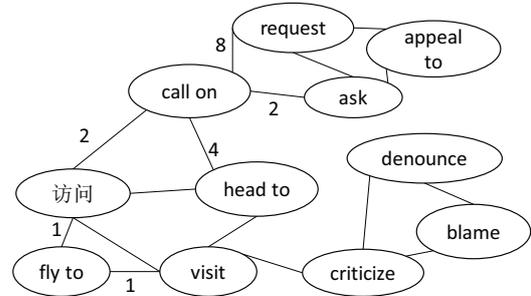


Figure 3: Network of relations. Edges indicate that the relations have a non-empty support intersection, and edge labels show the size of the intersection.

We found that the connectivity among similar relations is more important than the strength of the similarity. For example, as shown in Figure 3, *visit* is connected to most of the *visit*-relations such as *head to*, 访问. Although *visit* is connected to *criticize*, *visit* is not connected to other *criticize*-relations such as *denounce* and *blame*, whereas *criticize*, *denounce*, and *blame* are inter-

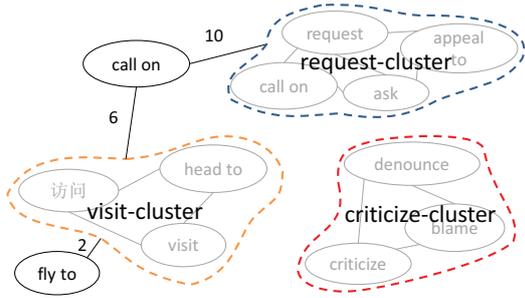


Figure 4: Relation clusters and a few individual relations. Edge labels show the size of the intersection.

connected. To exploit this feature, we use a random walk-based graph clustering method.

Formally, we use Markov clustering (Van Dongen, 2000) on a graph  $G = (V, E)$  of relations, where  $V = R_E \cup R_C$  is a set of all English and Chinese relations. An edge  $(r^i, r^j)$  indicates that two relations in any languages are similar, and its weight is quantified by a sigmoid function on a linear transformation of  $|H(r^i, r^j)|$  that was empirically found to produce good results.

Each resultant cluster forms a set of bilingual similar relations,  $c = \{r^{c1}, \dots, r^{cM}\}$ , such as *visit-cluster*, which consists of *visit*, *head to*, and *访问* in Figure 4. However, this cluster may not contain all similar relations. A relation may have multiple meanings (e.g., *call on*) so it can be clustered to another cluster, or a relation might not be clustered when its support set is too small (e.g., *fly to*). For such relations, rather than assigning zero similarity to *visit*-relations, we compute a cluster membership function based on support pairs of the cluster members and the target relation, and then formulate a pairwise relation translation score.

Formally, we learn the membership function of a relation  $r$  to a cluster  $c$  using support vector regression (Joachims, 1999) with the following features based on the support set of cluster  $c$ ,  $H(c) = \bigcup_{r \in c} H(r)$ , and the support intersection of  $r$  and  $c$ ,  $H(r, c) = \bigcup_{r^* \in c} H(r, r^*)$ .

- $f^1(r, c) = |H(r, c)|/|H(r)|$ : This quantifies the degree of inclusion,  $H(c) \in H(r)$ .
- $f^2(r, c) = |H(r, c)|/|H(c)|$ : This quantifies the degree of inclusion,  $H(r) \in H(c)$ .
- $f^3(r, c) = |H_{\text{within}}(r, c)|/|H_{\text{within}}(c)|$ : This is a variation of  $f^2$  that considers only noun phrase pairs shared at least once by relations in  $c$ .

- $f^4(r, c) = |H_{\text{within}}(r, c)|/|H_{\text{shared}}(c)|$ : This is a variation of  $f^2$  that considers only noun phrase pairs shared at least once by any pair of relations.
- $f^5(r, c) = |\{r^* \in c | H(r, r^*) > 0\}|/|c|$ : This is the degree of connectivity to the cluster members.

where  $H_{\text{within}}(r, c) = \bigcup_{r^* \in c} H(r, c) \cap H(r, r^*)$ , the intersection, considering translation, of  $H(r)$  and noun phrase pairs shared at once by relations in  $c$ ,  $H_{\text{within}}(c) = \bigcup_{r^* \in c} H(r^*, c - \{r^*\})$ , and  $H_{\text{shared}}(c) = \bigcup_{r^* \in R_E \cup R_C} H(r^*, c)$ , the noun phrase pairs shared at once by any relations. The use of  $H_{\text{within}}$  and  $H_{\text{shared}}$  is based on the observation that a noun phrase pair that appear in only one relation tends to be an incorrectly chunked entity such as ‘World Trade’ from the ‘World Trade Organization’.

Based on this membership function  $S(r, c)$ , we compute pairwise relation similarity. We consider that two relations are similar if they have at least one cluster that the both relations belong to, which can be measured with  $S(r, c)$ . More formally, pairwise similarity of relations  $r^i$  and  $r^j$  is defined as

$$T_R(r^i, r^j) = \max_{c \in \mathcal{C}} S(r^i, c) \cdot S(r^j, c) \quad (4)$$

where  $\mathcal{C}$  is a set of all clusters.

#### 4.4 Statement-level Document Context Comparison

A brute-force statement matching approach often fails due to ambiguity created by ignoring context, and missing information in  $T_N$  or  $T_R$ . Therefore, we detect similar document pairs to boost the statement matching process. Unlike the previous approaches (e.g., bag-of-words), we focus on the relationships of entities within documents using the extracted statements.

Formally, we compute the similarity of two statements  $s_E = (x_E, r_E, y_E)$  and  $s_C = (x_C, r_C, y_C)$  in different languages as follows:

$$T_S(s_E, s_C) = T_N(x_E, x_C) T_R(r_E, r_C) T_N(y_E, y_C) \quad (5)$$

With this definition, we can find similar statements described with different vocabularies in different languages.

To compare a document pair, we use the following equation to measure the similarity of an

English document  $d_E^i$  and a Chinese document  $d_C^j$  based on their statements  $S_E^i$  and  $S_C^j$ , respectively:

$$T_D(d_E^i, d_C^j) = \frac{\sum_{(s_E, s_C) \in B} T_S(s_E^{i,r}, s_C^{j,r})}{|S_E^i| + |S_C^j| - |B|} \quad (6)$$

where  $B \subset S_E^i \times S_C^j$  is a greedy approximate solution of maximum bipartite matching (West, 1999) on a bipartite graph  $G_B = (V_B = (S_E^i, S_C^j), E_B)$  with edge weights that are defined by  $T_S$ . The maximum bipartite matching finds a subset of edges in  $S_E^i \times S_C^j$  that maximize the sum of the selected edge weights and that do not share a node as their anchor point.

#### 4.5 Iteration on $T_N$

In this section, we describe how we use the statement similarity function  $T_S$ , and the document similarity function  $T_D$  to improve and derive the next generation entity translation function  $T_N^{(t+1)}$ . We consider that a pair of an English entity  $e_E$  and a Chinese entity  $e_C$  are likely to indicate the same real world entity if they have 1) semantically similar relations to the same entity 2) under the same context. Formally, we define an increment function as follows.

$$\Delta T_N(e_E, e_C) = \sum_{d_E^i} \sum_{d_C^j} T_D(d^i, d^j) \max_{(s_E, s_C) \in B^*} T_S(s_E, s_C) \quad (7)$$

where  $B^*$  is a subset of  $B \subset S_E^i \times S_C^j$  such that the connected statements mention  $e_E$  and  $e_C$ , and  $B$  is the greedy approximate solution of maximum bipartite matching for the set  $S_E^i$  of statements of  $d_E^i$  and the set  $S_C^j$  of statements of  $d_C^j$ . In other words,  $B^*$  is a set of matching statement pairs mentioning the translation target  $e_E$  and  $e_C$  in the document pair. Then, we use the following equation to improve the original entity translation function.

$$T_N^{(t+1)}(e_E, e_C) = (1 - \lambda) \frac{\Delta T_N(e_E, e_C)}{\sum_{e_C^*} \Delta T_N(e_E, e_C^*)} + \lambda T_N(e_E, e_C) \quad (8)$$

where  $\lambda$  is a mixing parameter in  $[0, 1]$ . We set  $\lambda = 0.6$  in our experiments.

With this update, we obtain the improved NE translations considering the relations that an entity has to other entities under the same context to achieve higher precision.

## 5 Experiments

In this section, we present experimental settings and results of translating entity names using our methods compared with several baselines.

### 5.1 Data and Evaluation

We processed news articles for an entire year in 2008 by Xinhua news who publishes news in both English and Chinese, which were also used by Kim et al. (2011) and Shao and Ng (2004). The English corpus consists of 100,746 news articles, and the Chinese corpus consists of 88,031 news articles. The news corpora are not *parallel* but *comparable* corpora, with asymmetry of entities and relationship as the asymmetry in the number of documents also suggest. Examples of such locality in Xinhua news include the more extensive coverage of Chinese teams in the Olympics and domestic sports in the Chinese news. Our framework finds and leverages comparable parts from the corpora without document-content-external information such as time stamps. We also show that, under the decreasing comparability, our method retains higher MRR than the baselines.

We follow the evaluation procedures used by You et al. (2012) and Kim et al. (2011) to fairly and precisely compare the effectiveness of our methods with baselines. To measure performance, we use mean reciprocal rank (MRR) to evaluate a translation function  $T$ :

$$MRR(T) = \frac{1}{|Q|} \sum_{(u,v) \in Q} \frac{1}{rank_T(u,v)} \quad (9)$$

where  $Q$  is the set of gold English-Chinese translation pairs  $(u, v)$  and  $rank_T(u, v)$  is the rank of  $T(u, v)$  in  $\{T(u, w) | w \text{ is a Chinese entity}\}$ . In addition, we use precision, recall, and F1-score.

As gold translation pairs, we use the evaluation data used by You et al. (2012) with additional labels, especially for organizations. The labeling task is done by randomly selecting English entities and finding their Chinese translation from the Chinese corpus. We only use entities with translations that appear in the Chinese corpus. We present the evaluation results for persons and organizations to show the robustness of the methods. In total, we identified 490 English entities in the English news with Chinese translations in the Chinese news. Among the 490 entities, 221 NEs are persons and 52 NEs are organizations.

	Person			Organization				
	MRR	P.	R.	F1	MRR	P.	R.	F1
$T_N^{(2)}$	<b>0.80</b>	<b>0.81</b>	<b>0.79</b>	<b>0.80</b>	<b>0.53</b>	<b>0.56</b>	<b>0.52</b>	<b>0.54</b>
$T_N^{(1)}$	0.77	0.80	0.77	0.78	0.44	0.49	0.44	0.46
$T_{PH+P}^S$	0.73	0.70	0.67	0.69	0.14	0.17	0.04	0.06
$T_{PH+P}^M$	0.68	0.70	0.68	0.69	0.08	0.31	0.08	0.12
$T_{HB}$	0.71	0.59	0.59	0.59	0.37	0.29	0.29	0.29
$T_{Dict}$	0.09	1.00	0.09	0.17	1.00	0.17	0.17	0.30

Table 3: Evaluation results of the methods.

## 5.2 Baselines

We compare our methods with the following baselines.

- $T_{PH+P}^S$  (You et al., 2012) is a holistic method that uses a transliteration method as base translations, and then reinforces them to achieve higher quality. This method uses only a single type of entities to propagate the translation scores.
- $T_{PH+P}^M$  is the holistic method revised to use naive multi-type propagation that uses multiple types of entities to reinforce the translation scores.
- $T_{HB}$  is a linear combination of transliteration and semantic translation methods (Lam et al., 2007) tuned to achieve the highest MRR.
- $T_{Dict}$  is a dictionary-only method. This dictionary is used by both  $T_{HB}$  and  $T_N$ .

Only the translation pairs of scores above 0.35 are used for  $T_{PH+P}$  to maximize the F1-score to measure precision, recall and F1-score. For our method  $T_N^{(t)}$ , we use the result with  $(t) = 1$ , the seed translations, and  $(t) = 2$ , which means that only one pass of the whole framework is performed to improve the seed translation function. In addition, we use translation pairs with scores above 0.05 to measure precision, recall, and F1-score. Note that these thresholds do not affect MRRs.

## 5.3 NE Translation Results

We show the result of the quantitative evaluation in Table 3, where the highest values are boldfaced, except  $T_{Dict}$  which shows 1.00 precision because it is a manually created dictionary. For both the person and organization cases, our method  $T_N^{(2)}$  outperforms the state-of-the-art methods in terms

English name	$T_N^{(2)}$	$T_N^{(1)}$	$T_{HB}$
Mandelson	<b>曼德尔森</b> [ManDeErSen]	<b>曼德尔森</b> [ManDeErSen]	<b>曼德尔森</b> [ManDeErSen]
WTO	<b>世贸组织</b> [ShiMaoZuZhi]	上合组织 [ShangHeZuZhi]	巴解组织 [BaJieZuZhi]
White House	<b>白宫</b> [BaiGong]	加州 [JiaZhou]	加州 [JiaZhou]
Microsoft	<b>微软公司</b> [WeiRuanGongSi]	美国司法部 [MeiGuoSiFaBu]	米罗诺夫 [MiLuoNuoFu]

Table 4: Example translations from the different methods. Boldface indicates correct translations.

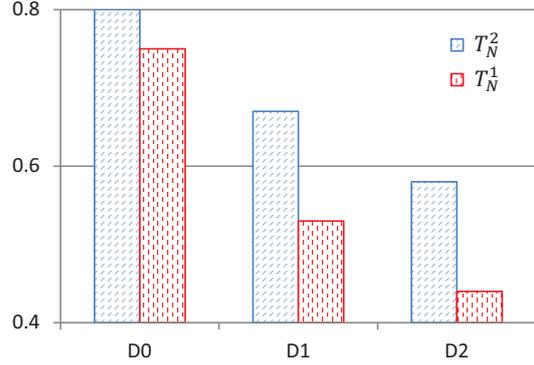


Figure 6: MRR with decreasing comparability.

of precision, recall, F1-score and MRR. With only one iteration of selective propagation, the seed translation is improved to achieve the 0.09 higher MRR.

The baselines show lower, but comparable MRRs and F1-scores for persons that mostly consist of transliterated cases. However, not all translations have phonetic similarity, especially organization names, as the low F1-score of  $T_{PH+P}^S$ , 0.06, for organizations suggests. The naive multi-type propagation  $T_{PH+P}^M$  shows decreased MRR for both persons and organizations compared to the single-type propagation  $T_{PH+P}^S$ , which shows a negative influence of diverse relation semantics of entities of different types.  $T_{HB}$  achieves a better MRR than  $T_{PH+P}$  due to the semantic translation of organization names. However, despite the increased recall of  $T_{HB}$  over that of  $T_{Dict}$ , the precision of  $T_{HB}$  is unsatisfactory because  $T_{HB}$  maps abbreviated names such as ‘WTO’ with other NEs. On the other hand, our method achieves the highest MRR and precision in both the person and organization categories.

As shown in Table 4,  $T_{HB}$  translates ‘WTO’ inaccurately, linking it to an incorrect organization ‘巴解组织’ (Palestine Liberation Organization).

The European Union (EU) Trade Commissioner (1) Peter Mandelson traveled to Moscow on Thursday for talks on ... Mandelson said it is a priority to see (2) Russia join the WTO, ...

欧盟贸易委员 (1) 彼得曼德尔森14日启程前往莫斯科, ... 德尔森在行前发表的声明中说, (2) 俄罗斯加入世贸组织是欧盟优先考虑的事项之一, ...



Figure 5: Example of similar document pairs.

Moreover, the use of the corpora by  $T_N^{(1)}$  could not fix this problem, and it finds another organization related to trade, ‘上合组织’ (Shanghai Cooperation Organization). In contrast, our selective propagation method  $T_N^{(2)}$ , which uses the wrong seed translation by  $T_N^{(1)}$ , ‘上合组织’ (Shanghai Cooperation Organization), successfully translates the WTO using statements such as (Russia, join, WTO), and its corresponding Chinese statement (俄罗斯, 加入, 世贸组织). Similarly, both the baseline  $T_{HB}$  and the seed translation  $T_N^{(1)}$  matched *Microsoft* to incorrect Chinese entities that are phonetically similar as indicated by the underlined text. In contrast,  $T_N^{(2)}$  finds the correct translation despite the phonetic dissimilarity.

#### 5.4 NE Translation Results with Low Corpus Comparability

We tested the methods using less comparable data to evaluate the robustness with the following derived datasets:

- D0: All news articles are used.
- D1: January-December English and July-December Chinese articles are used.
- D2: April-September English and July-December Chinese articles are used.

Figure 6 shows the MRR comparisons of our method  $T_N^{(2)}$  and  $T_N^{(1)}$  on all test entities. Because the commonly appearing NEs are decreasing, the performance decline is inevitable. However, we can see that the MRR of the seed translation method drops significantly on D1 and D2, whereas our method shows 0.14 higher MRR for both cases.

#### 5.5 Similar Documents

In this section, we show an example of similar documents in Figure 5. Both articles describe the same event about the visit of Mandelson to Moscow for the discussion on the joining of Russia to the WTO. The extracted statements are the exact translations of each corresponding part as indicated by the arrows. We stress this is an extreme case for illustration, where the two sentences are almost an exact translation, except for a minor asymmetry involving the date (Thursday in English, and 14th in Chinese). In most similar documents, the asymmetry is more significant. The seed translation score  $T_N^1(\text{WTO}, \text{世贸组织})$  is not enough to match the entities. However, the context similarity, due to other similar statements such as (1), allows us to match (2). This match helps translation of ‘WTO’ by inspecting the organization that Russia considers to join in both documents.

#### 6 Conclusions

This paper proposed a bootstrapping approach for entity translation using multilingual relational clustering. Further, the proposed method could find similar document pairs by comparing statements to enable us to focus on comparable parts of evidence. We validated the quality of our approach using real-life English and Chinese corpora, and its performance significantly exceeds that of previous approaches.

#### Acknowledgment

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea and Microsoft Research, under IT/SW Creative research program supervised by the NIPA (National IT Industry Promotion Agency). (NIPA-2012-H0503-12-1036).

## References

- Donghui Feng. 2004. A new approach for english-chinese named entity alignment. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP*, pages 372–379.
- Long Jiang, Shiquan Yang, Ming Zhou, Xiaohua Liu, and Qingsheng Zhu. 2009. Mining bilingual data from the web with adaptively learnt patterns. In *Joint Conference of the ACL and the IJCNLP*, pages 870–878, Stroudsburg, PA, USA.
- T. Joachims. 1999. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge, MA, USA.
- Seokhwan Kim, Minwoo Jeong, Jonghoon Lee, and Gary Geunbae Lee. 2010. A cross-lingual annotation projection approach for relation detection. In *COLING*, pages 564–571, Stroudsburg, PA, USA.
- Jinhan Kim, Long Jiang, Seung-won Hwang, Young-In Song, and Ming Zhou. 2011. Mining entity translations from comparable corpora: a holistic graph mapping approach. In *CIKM*, pages 1295–1304, New York, NY, USA.
- Jinhan Kim, Seung won Hwang, Long Jiang, Young-In Song, and Ming Zhou. 2012. Entity translation mining from comparable corpora: Combining graph mapping with corpus latent features. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints).
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Alexandre Klementiev and Dan Roth. 2006. Named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 82–88, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Comput. Linguist.*, 24(4):599–612, December.
- Julian Kupiec. 1993. An algorithm for finding noun phrase correspondences in bilingual corpora. In *ACL*, pages 17–22, Stroudsburg, PA, USA.
- Wai Lam, Shing-Kit Chan, and Ruizhang Huang. 2007. Named entity translation matching and learning: With application for mining unseen translations. *ACM Trans. Inf. Syst.*, 25(1), February.
- Dekang Lin, Shaojun Zhao, Benjamin Van Durme, and Marius Pasca. 2008. Mining parenthetical translations from the web by word alignment. In *ACL*.
- Thahir Mohamed, Estevam Hruschka, and Tom Mitchell. 2011. Discovering relations between noun categories. In *EMNLP*, pages 1447–1455, Edinburgh, Scotland, UK., July.
- Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *EMNLP*.
- Patrick Schone, Tim Allison, Chris Giannella, and Craig Pfeifer. 2011. Bootstrapping multilingual relation discovery using english wikipedia and wikimedia-induced entity extraction. In *ICTAI*, pages 944–951, Washington, DC, USA.
- Li Shao and Hwee Tou Ng. 2004. Mining new word translations from comparable corpora. In *COLING*, Stroudsburg, PA, USA.
- S. Van Dongen. 2000. *Graph Clustering by Flow Simulation*. Ph.D. thesis, University of Utrecht, The Netherlands.
- Stephen Wan and Cornelia Maria Verspoor. 1998. Automatic english-chinese name transliteration for development of multilingual resources. In *ACL*, pages 1352–1356, Stroudsburg, PA, USA.
- Douglas Brent West. 1999. *Introduction to graph theory (2nd edition)*. Prentice Hall.
- Fan Yang, Jun Zhao, and Kang Liu. 2009. A chinese-english organization name translation system using heuristic web mining and asymmetric alignment. In *Joint Conference of the ACL and the IJCNLP*, pages 387–395, Stroudsburg, PA, USA.
- Gae-won You, Seung-won Hwang, Young-In Song, Long Jiang, and Zaiqing Nie. 2010. Mining name translations from entity graph mapping. In *EMNLP*, pages 430–439, Stroudsburg, PA, USA.
- Gae-Won You, Seung-Won Hwang, Young-In Song, Long Jiang, and Zaiqing Nie. 2012. Efficient entity translation mining: A parallelized graph alignment approach. *ACM Trans. Inf. Syst.*, 30(4):25:1–25:23, November.

# Transfer Learning Based Cross-lingual Knowledge Extraction for Wikipedia

Zhigang Wang<sup>†</sup>, Zhixing Li<sup>†</sup>, Juanzi Li<sup>†</sup>, Jie Tang<sup>†</sup>, and Jeff Z. Pan<sup>‡</sup>

<sup>†</sup> Tsinghua National Laboratory for Information Science and Technology  
DCST, Tsinghua University, Beijing, China

{wzhigang, zhxli, ljz, tangjie}@keg.cs.tsinghua.edu.cn

<sup>‡</sup> Department of Computing Science, University of Aberdeen, Aberdeen, UK  
jeff.z.pan@abdn.ac.uk

## Abstract

Wikipedia infoboxes are a valuable source of structured knowledge for global knowledge sharing. However, infobox information is very incomplete and imbalanced among the Wikipedias in different languages. It is a promising but challenging problem to utilize the rich structured knowledge from a source language Wikipedia to help complete the missing infoboxes for a target language.

In this paper, we formulate the problem of cross-lingual knowledge extraction from multilingual Wikipedia sources, and present a novel framework, called WikiCiKE, to solve this problem. An instance-based transfer learning method is utilized to overcome the problems of topic drift and translation errors. Our experimental results demonstrate that WikiCiKE outperforms the monolingual knowledge extraction method and the translation-based method.

## 1 Introduction

In recent years, the automatic knowledge extraction using Wikipedia has attracted significant research interest in research fields, such as the semantic web. As a valuable source of structured knowledge, Wikipedia infoboxes have been utilized to build linked open data (Suchanek et al., 2007; Bollacker et al., 2008; Bizer et al., 2008; Bizer et al., 2009), support next-generation information retrieval (Hotho et al., 2006), improve question answering (Bouma et al., 2008; Fernández et al., 2009), and other aspects of data exploitation (McIlraith et al., 2001; Volkel et al., 2006; Hogan et al., 2011) using semantic web standards, such as RDF (Pan and Horrocks, 2007;

Heino and Pan, 2012) and OWL (Pan and Horrocks, 2006; Pan and Thomas, 2007; Fokoue et al., 2012), and their reasoning services.

However, most infoboxes in different Wikipedia language versions are missing. Figure 1 shows the statistics of article numbers and infobox information for six major Wikipedias. Only 32.82% of the articles have infoboxes on average, and the numbers of infoboxes for these Wikipedias vary significantly. For instance, the English Wikipedia has 13 times more infoboxes than the Chinese Wikipedia and 3.5 times more infoboxes than the second largest Wikipedia of German language.

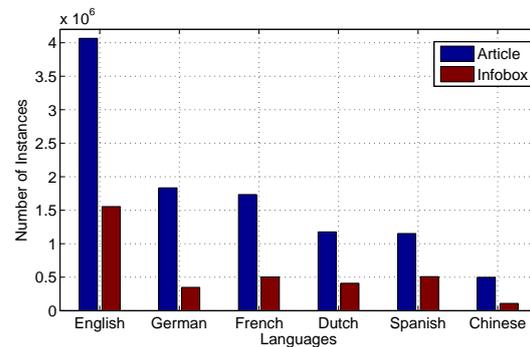


Figure 1: Statistics for Six Major Wikipedias.

To solve this problem, KYLIN has been proposed to extract the missing infoboxes from unstructured article texts for the English Wikipedia (Wu and Weld, 2007). KYLIN performs well when sufficient training data are available, and such techniques as shrinkage and retraining have been used to increase recall from English Wikipedia's long tail of sparse infobox classes (Weld et al., 2008; Wu et al., 2008). The extraction performance of KYLIN is limited by the number of available training samples.

Due to the great imbalance between different Wikipedia language versions, it is difficult to gather sufficient training data from a single Wikipedia. Some translation-based cross-lingual knowledge

extraction methods have been proposed (Adar et al., 2009; Bouma et al., 2009; Adafre and de Rijke, 2006). These methods concentrate on translating existing infoboxes from a richer source language version of Wikipedia into the target language. The recall of new target infoboxes is highly limited by the number of equivalent cross-lingual articles and the number of existing source infoboxes. Take Chinese-English<sup>1</sup> Wikipedias as an example: current translation-based methods only work for 87,603 Chinese Wikipedia articles, 20.43% of the total 428,777 articles. Hence, the challenge remains: how could we supplement the missing infoboxes for the rest 79.57% articles?

On the other hand, the numbers of existing infobox attributes in different languages are highly imbalanced. Table 1 shows the comparison of the numbers of the articles for the attributes in template PERSON between English and Chinese Wikipedia. Extracting the missing value for these attributes, such as *awards*, *weight*, *influences* and *style*, inside the single Chinese Wikipedia is intractable due to the rarity of existing Chinese attribute-value pairs.

Attribute	en	zh	Attribute	en	zh
<i>name</i>	82,099	1,486	<i>awards</i>	2,310	38
<i>birth date</i>	77,850	1,481	<i>weight</i>	480	12
<i>occupation</i>	66,768	1,279	<i>influences</i>	450	6
<i>nationality</i>	20,048	730	<i>style</i>	127	1

Table 1: The Numbers of Articles in TEMPLATE PERSON between English(en) and Chinese(zh).

In this paper, we have the following hypothesis: *one can use the rich English (auxiliary) information to assist the Chinese (target) infobox extraction*. In general, we address the problem of cross-lingual knowledge extraction by using the imbalance between Wikipedias of different languages. For each attribute, we aim to learn an extractor to find the missing value from the unstructured article texts in the target Wikipedia by using the rich information in the source language. Specifically, we treat this cross-lingual information extraction task as a transfer learning-based binary classification problem.

The contributions of this paper are as follows:

1. We propose a transfer learning-based cross-lingual knowledge extraction framework

<sup>1</sup>Chinese-English denotes the task of Chinese Wikipedia infobox completion using English Wikipedia

called **WikiCiKE**. The extraction performance for the target Wikipedia is improved by using rich infoboxes and textual information in the source language.

2. We propose the TrAdaBoost-based extractor training method to avoid the problems of topic drift and translation errors of the source Wikipedia. Meanwhile, some language-independent features are introduced to make WikiCiKE as general as possible.
3. Chinese-English experiments for four typical attributes demonstrate that WikiCiKE outperforms both the monolingual extraction method and current translation-based method. The increases of 12.65% for precision and 12.47% for recall in the template named person are achieved when only 30 target training articles are available.

The rest of this paper is organized as follows. Section 2 presents some basic concepts, the problem formalization and the overview of WikiCiKE. In Section 3, we propose our detailed approaches. We present our experiments in Section 4. Some related work is described in Section 5. We conclude our work and the future work in Section 6.

## 2 Preliminaries

In this section, we introduce some basic concepts regarding Wikipedia, formally defining the key problem of cross-lingual knowledge extraction and providing an overview of the WikiCiKE framework.

### 2.1 Wiki Knowledge Base and Wiki Article

We consider each language version of Wikipedia as a *wiki knowledge base*, which can be represented as  $K = \{a_i\}_{i=1}^p$ , where  $a_i$  is a disambiguated article in  $K$  and  $p$  is the size of  $K$ .

Formally we define a *wiki article*  $a \in K$  as a 5-tuple  $a = (\textit{title}, \textit{text}, \textit{ib}, \textit{tp}, C)$ , where

- *title* denotes the title of the article  $a$ ,
- *text* denotes the unstructured text description of the article  $a$ ,
- *ib* is the infobox associated with  $a$ ; specifically,  $\textit{ib} = \{(\textit{attr}_i, \textit{value}_i)\}_{i=1}^q$  represents the list of attribute-value pairs for the article  $a$ ,

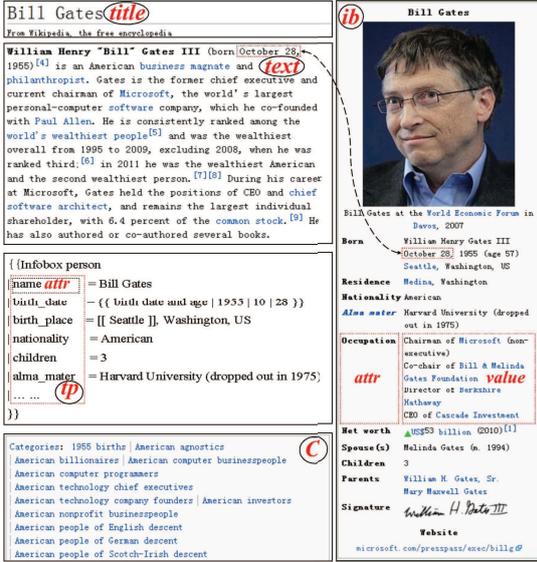


Figure 2: Simplified Article of “Bill Gates”.

- $tp = \{attr_i\}_{i=1}^r$  is the infobox template associated with  $ib$ , where  $r$  is the number of attributes for one specific template, and
- $C$  denotes the set of categories to which the article  $a$  belongs.

Figure 2 gives an example of these five important elements concerning the article named “Bill Gates”.

In what follows, we will use named subscripts, such as  $a_{Bill\ Gates}$ , or index subscripts, such as  $a_i$ , to refer to one particular instance interchangeably. We will use “name in TEMPLATE PERSON” to refer to the attribute  $attr_{name}$  in the template  $tp_{PERSON}$ . In this cross-lingual task, we use the source ( $S$ ) and target ( $T$ ) languages to denote the languages of auxiliary and target Wikipedias, respectively. For example,  $K_S$  indicates the source wiki knowledge base, and  $K_T$  denotes the target wiki knowledge base.

## 2.2 Problem Formulation

Mining new infobox information from unstructured article texts is actually a multi-template, multi-slot information extraction problem. In our task, each template represents an infobox template and each slot denotes an attribute. In the WikiCiKE framework, for each attribute  $attr_T$  in an infobox template  $tp_T$ , we treat the task of missing value extraction as a binary classification problem. It predicts whether a particular word (token) from the article  $text$  is the extraction target (Finn and Kushmerick, 2004; Lafferty et al., 2001).

Given an attribute  $attr_T$  and an instance (word/token)  $x_i$ ,  $X_S = \{x_i\}_{i=1}^n$  and  $X_T = \{x_i\}_{i=n+1}^{n+m}$  are the sets of instances (words/tokens) in the source and the target language respectively.  $x_i$  can be represented as a feature vector according to its context. Usually, we have  $n \gg m$  in our setting, with much more attributes in the source than those in the target. The function  $g : X \mapsto Y$  maps the instance from  $X = X_S \cup X_T$  to the true label of  $Y = \{0, 1\}$ , where 1 represents the extraction target (positive) and 0 denotes the background information (negative). Because the number of target instances  $m$  is inadequate to train a good classifier, we combine the source and target instances to construct the training data set as  $TD = TD_S \cup TD_T$ , where  $TD_S = \{x_i, g(x_i)\}_{i=1}^n$  and  $TD_T = \{x_i, g(x_i)\}_{i=n+1}^{n+m}$  represent the source and target training data, respectively.

Given the combined training data set  $TD$ , our objective is to estimate a hypothesis  $f : X \mapsto Y$  that minimizes the prediction error on testing data in the target language. Our idea is to determine the useful part of  $TD_S$  to improve the classification performance in  $TD_T$ . We view this as a transfer learning problem.

## 2.3 WikiCiKE Framework

WikiCiKE learns an extractor for a given attribute  $attr_T$  in the target Wikipedia. As shown in Figure 3, WikiCiKE contains four key components: (1) **Automatic Training Data Generation**: given the target attribute  $attr_T$  and two wiki knowledge bases  $K_S$  and  $K_T$ , WikiCiKE first generates the training data set  $TD = TD_S \cup TD_T$  automatically. (2) **WikiCiKE Training**: WikiCiKE uses a transfer learning-based classification method to train the classifier (extractor)  $f : X \mapsto Y$  by using  $TD_S \cup TD_T$ . (3) **Template Classification**: WikiCiKE then determines proper candidate articles which are suitable to generate the missing value of  $attr_T$ . (4) **WikiCiKE Extraction**: given a candidate article  $a$ , WikiCiKE uses the learned extractor  $f$  to label each word in the  $text$  of  $a$ , and generate the extraction result in the end.

## 3 Our Approach

In this section, we will present the detailed approaches used in WikiCiKE.

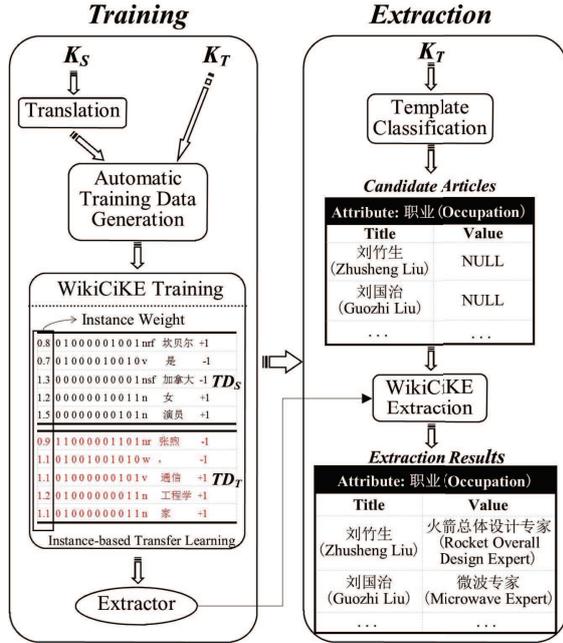


Figure 3: WikiCiKE Framework.

### 3.1 Automatic Training Data Generation

To generate the training data for the target attribute  $attr_T$ , we first determine the equivalent cross-lingual attribute  $attr_S$ . Fortunately, some templates in non-English Wikipedia (e.g. Chinese Wikipedia) explicitly match their attributes with their counterparts in English Wikipedia. Therefore, it is convenient to align the cross-lingual attributes using English Wikipedia as bridge. For attributes that can not be aligned in this way, currently we manually align them. The manual alignment is worthwhile because thousands of articles belong to the same template may benefit from it and at the same time it is not very costly. In Chinese Wikipedia, the top 100 templates have covered nearly 80% of the articles which have been assigned a template.

Once the aligned attribute mapping  $attr_T \leftrightarrow attr_S$  is obtained, we collect the articles from both  $K_S$  and  $K_T$  containing the corresponding  $attr$ . The collected articles from  $K_S$  are translated into the target language. Then, we use a uniform automatic method, which primarily consists of word labeling and feature vector generation, to generate the training data set  $TD = \{(x, g(x))\}$  from these collected articles.

For each collected article  $a = \{title, text, ib, tp, C\}$  and its *value* of  $attr$ , we can automatically label each word  $x$  in *text* according to whether  $x$  and its neighbors are

contained by the *value*. The *text* and *value* are processed as bags of words  $\{x\}_{text}$  and  $\{x\}_{value}$ . Then for each  $x_i \in \{x\}_{text}$  we have:

$$g(x_i) = \begin{cases} 1 & x_i \in \{x\}_{value}, |\{x\}_{value}| = 1 \\ 1 & x_{i-1}, x_i \in \{x\}_{value} \text{ OR } x_i, x_{i+1} \in \{x\}_{value}, \\ & |\{x\}_{value}| > 1 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

After the word labeling, each instance (word/token) is represented as a feature vector. In this paper, we propose a general feature space that is suitable for most target languages. As shown in Table 2, we classify the features used in WikiCiKE into three categories: format features, POS tag features and token features.

Category	Feature	Example
Format feature	First token of sentence	你好, 世界! <i>Hello World!</i>
	In first half of sentence	你好, 世界! <i>Hello World!</i>
	Starts with two digits	12月31日 31 <sup>th</sup> Dec.
	Starts with four digits	1999年 夏天 1999's summer
	Contains a cash sign	10 ¥ or 10 \$
	Contains a percentage symbol	10%
	Stop words	的, 地, 这... of, the, a, an
	Pure number	365
	Part of an anchor text	电影导演 <i>Movie Director</i>
	Begin of an anchor text	游戏设计师 <i>Game Designer</i>
POS tag features	POS tag of current token	
	POS tags of previous 5 tokens	
	POS tags of next 5 tokens	
Token features	Current token	
	Previous 5 tokens	
	Next 5 tokens	
	Is current token contained by title	
	Is one of previous 5 tokens contained by title	

Table 2: Feature Definition.

The target training data  $TD_T$  is directly generated from articles in the target language Wikipedia. Articles from the source language Wikipedia are translated into the target language in advance and then transformed into training data  $TD_S$ . In next section, we will discuss how to train an extractor from  $TD = TD_S \cup TD_T$ .

### 3.2 WikiCiKE Training

Given the attribute  $attr_T$ , we want to train a classifier  $f : X \mapsto Y$  that can minimize the prediction

error for the testing data in the target language. Traditional machine learning approaches attempt to determine  $f$  by minimizing some loss function  $L$  on the prediction  $f(x)$  for the training instance  $x$  and its real label  $g(x)$ , which is

$$\hat{f} = \operatorname{argmin}_{f \in \Theta} \sum L(f(x), g(x)) \text{ where } (x, g(x)) \in TD_T \quad (2)$$

In this paper, we use TrAdaBoost (Dai et al., 2007), which is an instance-based transfer learning algorithm that was first proposed by Dai to find  $\hat{f}$ . TrAdaBoost requires that the source training instances  $X_S$  and target training instances  $X_T$  be drawn from the same feature space. In WikiCiKE, the source articles are translated into the target language in advance to satisfy this requirement. Due to the topic drift problem and translation errors, the joint probability distribution  $P_S(x, g(x))$  is not identical to  $P_T(x, g(x))$ . We must adjust the source training data  $TD_S$  so that they fit the distribution on  $TD_T$ . TrAdaBoost iteratively updates the weights of all training instances to optimize the prediction error. Specifically, the weight-updating strategy for the source instances is decided by the loss on the target instances.

For each  $t = 1 \sim T$  iteration, given a weight vector  $\mathbf{p}_t$  normalized from  $\mathbf{w}_t$  ( $\mathbf{w}_t$  is the weight vector before normalization), we call a basic classifier  $F$  that can address weighted instances and then find a hypothesis  $f$  that satisfies

$$\hat{f}_t = \operatorname{argmin}_{f \in \Theta_F} \sum L(\mathbf{p}_t, f(x), g(x)) \quad (3)$$

$$(x, g(x)) \in TD_S \cup TD_T$$

Let  $\epsilon_t$  be the prediction error of  $\hat{f}_t$  at the  $t^{\text{th}}$  iteration on the target training instances  $TD_T$ , which is

$$\epsilon_t = \frac{1}{\sum_{k=n+1}^{n+m} w_k^t} \times \sum_{k=n+1}^{n+m} (w_k^t \times |\hat{f}_t(x_k) - y_k|) \quad (4)$$

With  $\epsilon_t$ , the weight vector  $\mathbf{w}_t$  is updated by the function:

$$\mathbf{w}_{t+1} = h(\mathbf{w}_t, \epsilon_t) \quad (5)$$

The weight-updating strategy  $h$  is illustrated in Table 3.

Finally, a final classifier  $\hat{f}$  can be obtained by combining  $\hat{f}_{T/2} \sim \hat{f}_T$ .

TrAdaBoost has a convergence rate of  $O(\sqrt{\ln(n/N)})$ , where  $n$  and  $N$  are the number of source samples and number of maximum iterations respectively.

		TrAdaBoost	AdaBoost
Target samples	+	$w_t$	$w_t$
	-	$w_t \times \beta_t^{-1}$	$w_t \times \beta_t^{-1}$
Source samples	+	$w_t \times \beta^{-1}$	No source training
	-	$w_t \times \beta$	sample available

+: correctly labelled      -: miss-labelled  
 $w_t$ : weight of an instance at the  $t^{\text{th}}$  iteration  
 $\beta_t = \epsilon_t \times (1 - \epsilon_t)$   
 $\beta = 1/(1 + \sqrt{2 \ln nT})$

Table 3: Weight-updating Strategy of TrAdaBoost.

### 3.3 Template Classification

Before using the learned classifier  $f$  to extract missing infobox value for the target attribute  $attr_T$ , we must select the correct articles to be processed. For example, the article  $a_{New York}$  is not a proper article for extracting the missing value of the attribute  $attr_{birth\_day}$ .

If  $a$  already has an incomplete infobox, it is clear that the correct  $tp$  is the template of its own infobox  $ib$ . For those articles that have no infoboxes, we use the classical 5-nearest neighbor algorithm to determine their templates (Roussopoulos et al., 1995) using their category labels, outlinks, inlinks as features (Wang et al., 2012). Our classifier achieves an average precision of 76.96% with an average recall of 63.29%, and can be improved further. In this paper, we concentrate on the WikiCiKE training and extraction components.

### 3.4 WikiCiKE Extraction

Given an article  $a$  determined by template classification, we generate the missing *value* of *attr* from the corresponding *text*. First, we turn the *text* into a word sequence and compute the feature vector for each word based on the feature definition in Section 3.1. Next we use  $f$  to label each word, and we get a labeled sequence  $text^l$  as  $text^l = \{x_1^{f(x_1)} \dots x_{i-1}^{f(x_{i-1})} x_i^{f(x_i)} x_{i+1}^{f(x_{i+1})} \dots x_n^{f(x_n)}\}$  where the superscript  $f(x_i) \in \{0, 1\}$  represents the positive or negative label by  $f$ . After that, we extract the adjacent positive tokens in *text* as the predict value. In particular, the longest positive token sequence and the one that contains other positive token sequences are preferred in extraction. E.g., a positive sequence “comedy movie director” is preferred to a shorter sequence “movie director”.

## 4 Experiments

In this section, we present our experiments to evaluate the effectiveness of WikiCiKE, where we focus on the Chinese-English case; in other words, the target language is Chinese and the source language is English. It is part of our future work to try other language pairs which two Wikipedias of these languages are imbalanced in infobox information such as English-Dutch.

### 4.1 Experimental Setup

#### 4.1.1 Data Sets

Our data sets are from Wikipedia dumps<sup>2</sup> generated on April 3, 2012. For each attribute, we collect both labeled articles (articles that contain the corresponding attribute *attr*) and unlabeled articles in Chinese. We split the labeled articles into two subsets  $A_T$  and  $A_{test}$  ( $A_T \cap A_{test} = \emptyset$ ), in which  $A_T$  is used as target training articles and  $A_{test}$  is used as the first testing set. For the unlabeled articles, represented as  $A'_{test}$ , we manually label their infoboxes with their texts and use them as the second testing set. For each attribute, we also collect a set of labeled articles  $A_S$  in English as the source training data. Our experiments are performed on four attributes, which are *occupation*, *nationality*, *alma mater* in TEMPLATE PERSON, and *country* in TEMPLATE FILM. In particular, we extract values from the first two paragraphs of the texts because they usually contain most of the valuable information. The details of data sets on these attributes are given in Table 4.

Attribute	$ A_S $	$ A_T $	$ A_{test} $	$ A'_{test} $
<i>occupation</i>	1,000	500	779	208
<i>alma mater</i>	1,000	200	215	208
<i>nationality</i>	1,000	300	430	208
<i>country</i>	1,000	500	1,000	—

$|A|$ : the number of articles in  $A$

Table 4: Data Sets.

#### 4.1.2 Comparison Methods

We compare our WikiCiKE method with two different kinds of methods, the monolingual knowledge extraction method and the translation-based method. They are implemented as follows:

1. **KE-Mon** is the monolingual knowledge extractor. The difference between WikiCiKE and KE-Mon is that KE-Mon only uses the Chinese training data.

<sup>2</sup><http://dumps.wikimedia.org/>

2. **KE-Tr** is the translation-based extractor. It obtains the *values* by two steps: finding their counterparts (if available) in English using Wikipedia cross-lingual links and attribute alignments, and translating them into Chinese.

We conduct two series of evaluation to compare WikiCiKE with KE-Mon and KE-Tr, respectively.

1. We compare WikiCiKE with KE-Mon on the first testing data set  $A_{test}$ , where most values can be found in the articles' texts in those labeled articles, in order to demonstrate the performance improvement by using cross-lingual knowledge transfer.
2. We compare WikiCiKE with KE-Tr on the second testing data set  $A'_{test}$ , where the existences of values are not guaranteed in those randomly selected articles, in order to demonstrate the better recall of WikiCiKE.

For implementation details, the *weighted-SVM* is used as the basic learner  $f$  both in WikiCiKE and KE-Mon (Zhang et al., 2009), and Baidu Translation API<sup>3</sup> is used as the translator both in WikiCiKE and KE-Tr. The Chinese texts are pre-processed using ICTCLAS<sup>4</sup> for word segmentation.

#### 4.1.3 Evaluation Metrics

Following Lavelli's research on evaluation of information extraction (Lavelli et al., 2008), we perform evaluation as follows.

1. We evaluate each *attr* separately.
2. For each *attr*, there is exactly one *value* extracted.
3. No alternative occurrence of real *value* is available.
4. The overlap ratio is used in this paper rather than "exactly matching" and "containing".

Given an extracted *value*  $v' = \{w'\}$  and its corresponding real *value*  $v = \{w\}$ , two measurements for evaluating the overlap ratio are defined: **recall**: the rate of matched tokens w.r.t. the real *value*. It can be calculated using

$$R(v', v) = \frac{|v \cap v'|}{|v|}$$

<sup>3</sup><http://openapi.baidu.com/service>

<sup>4</sup><http://www.ictclas.org/>

**precision**: the rate of matched tokens w.r.t. the extracted *value*. It can be calculated using

$$P(v', v) = \frac{|v \cap v'|}{|v'|}$$

We use the average of these two measures to evaluate the performance of our extractor as follows:

$$R = \text{avg}(R_i(v', v)) \quad a_i \in A_{\text{test}}$$

$$P = \text{avg}(P_i(v', v)) \quad a_i \in A_{\text{test}} \text{ and } v_i' \neq \emptyset$$

The *recall* and *precision* range from 0 to 1 and are first calculated on a single instance and then averaged over the testing instances.

## 4.2 Comparison with KE-Mon

In these experiments, WikiCiKE trains extractors on  $A_S \cup A_T$ , and KE-Mon trains extractors just on  $A_T$ . We incrementally increase the number of target training articles from 10 to 500 (if available) to compare WikiCiKE with KE-Mon in different situations. We use the first testing data set  $A_{\text{test}}$  to evaluate the results.

Figure 4 and Table 5 show the experimental results on TEMPLATE PERSON and FILM. We can see that WikiCiKE outperforms KE-Mon on all three attributions especially when the number of target training samples is small. Although the *recall* for *alma mater* and the *precision* for *nationality* of WikiCiKE are lower than KE-Mon when only 10 target training articles are available, WikiCiKE performs better than KE-Mon if we take into consideration both *precision* and *recall*.

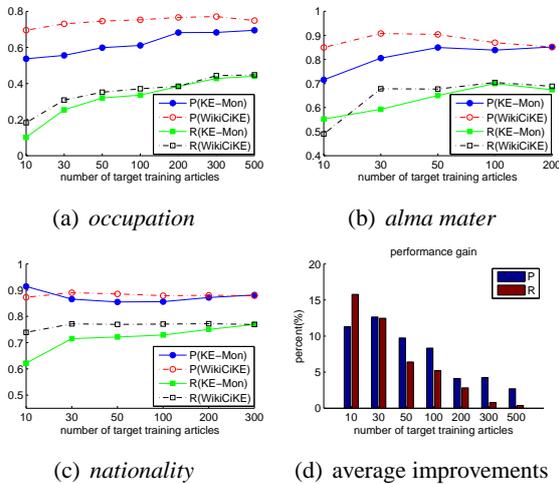


Figure 4: Results for TEMPLATE PERSON.

Figure 4(d) shows the average improvements yielded by WikiCiKE w.r.t KE-Mon on TEMPLATE PERSON. We can see that WikiCiKE yields significant improvements when only a few articles are available in target language and the improvements tend to decrease as the number of target articles is increased. In this case, the articles in the target language are sufficient to train the extractors alone.

#	KE-Mon		WikiCiKE	
	P	R	P	R
10	81.1%	63.8%	<b>90.7%</b>	66.3%
30	78.8%	64.5%	87.5%	69.4%
50	80.7%	66.6%	87.7%	<b>72.3%</b>
100	82.8%	68.2%	87.8%	72.1%
200	83.6%	70.5%	87.1%	73.2%
300	85.2%	72.0%	89.1%	76.2%
500	86.2%	73.4%	88.7%	75.6%

# Number of the target training articles.

Table 5: Results for *country* in TEMPLATE FILM.

## 4.3 Comparison with KE-Tr

We compare WikiCiKE with KE-Tr on the second testing data set  $A'_{\text{test}}$ .

From Table 6 it can be clearly observed that WikiCiKE significantly outperforms KE-Tr both in *precision* and *recall*. The reasons why the recall of KE-Tr is extremely low are two-fold. First, because of the limit of cross-lingual links and infoboxes in English Wikipedia, only a very small set of values is found by KE-Tr. Furthermore, many values obtained using the translator are incorrect because of translation errors. WikiCiKE uses translators too, but it has better tolerance to translation errors because the extracted value is from the target article texts instead of the output of translators.

Attribute	KE-Tr		WikiCiKE	
	P	R	P	R
<i>occupation</i>	27.4%	3.40%	64.8%	26.4%
<i>nationality</i>	66.3%	4.60%	70.0%	<b>55.0%</b>
<i>alma mater</i>	66.7%	0.70%	<b>76.3%</b>	8.20%

Table 6: Results of WikiCiKE vs. KE-Tr.

## 4.4 Significance Test

We conducted a significance test to demonstrate that the difference between WikiCiKE and KE-Mon is significant rather than caused by statistical errors. As for the comparison between WikiCiKE and KE-Tr, significant improvements brought by

WikiCiKE can be clearly observed from Table 6 so there is no need for further significance test. In this paper, we use McNemar’s significance test (Dietterich and Thomas, 1998).

Table 7 shows the results of significance test calculated for the average on all tested attributes. When the number of target training articles is less than 100, the  $\chi$  is much less than 10.83 that corresponds to a significance level 0.001. It suggests that the chance that WikiCiKE is not better than KE-Mon is less than 0.001.

#	10	30	50	100	200	300	500
$\chi$	179.5	107.3	51.8	32.8	4.1	4.3	0.3

# Number of the target training articles.

Table 7: Results of Significance Test.

#### 4.5 Overall Analysis

As shown in above experiments, we can see that WikiCiKE outperforms both KE-Mon and KE-Tr. When only 30 target training samples are available, WikiCiKE reaches comparable performance of KE-Mon using 300-500 target training samples. Among all of the 72 attributes in TEMPLATE PERSON of Chinese Wikipedia, 39 (54.17%) and 55 (76.39%) attributes have less than 30 and 200 labeled articles respectively. We can see that WikiCiKE can save considerable human labor when no sufficient target training samples are available.

We also examined the errors by WikiCiKE and they can be categorized into three classes. For attribute *occupation* when 30 target training samples are used, there are 71 errors. The first category is caused by incorrect word segmentation (40.85%). In Chinese, there is no space between words so we need to segment them before extraction. The result of word segmentation directly decide the performance of extraction so it causes most of the errors. The second category is because of the incomplete infoboxes (36.62%). In evaluation of KE-Mon, we directly use the values in infoboxes as golden values, some of them are incomplete so the correct predicted values will be automatically judged as the incorrect in these cases. The last category is mismatched words (22.54%). The predicted value does not match the golden value or a part of it. In the future, we can improve the performance of WikiCiKE by polishing the word segmentation result.

## 5 Related Work

Some approaches of knowledge extraction from the open Web have been proposed (Wu et al., 2012; Yates et al., 2007). Here we focus on the extraction inside Wikipedia.

### 5.1 Monolingual Infobox Extraction

KYLIN is the first system to autonomously extract the missing infoboxes from the corresponding article texts by using a self-supervised learning method (Wu and Weld, 2007). KYLIN performs well when enough training data are available. Such techniques as shrinkage and retraining are proposed to increase the recall from English Wikipedia’s long tail of sparse classes (Wu et al., 2008; Wu and Weld, 2010). Different from Wu’s research, WikiCiKE is a cross-lingual knowledge extraction framework, which leverags rich knowledge in the other language to improve extraction performance in the target Wikipedia.

### 5.2 Cross-lingual Infobox Completion

Current translation based methods usually contain two steps: cross-lingual attribute alignment and value translation. The attribute alignment strategies can be grouped into two categories: cross-lingual link based methods (Bouma et al., 2009) and classification based methods (Adar et al., 2009; Nguyen et al., 2011; Aumüller et al., 2005; Adafre and de Rijke, 2006; Li et al., 2009). After the first step, the value in the source language is translated into the target language. E. Adar’s approach gives the overall precision of 54% and recall of 40% (Adar et al., 2009). However, recall of these methods is limited by the number of equivalent cross-lingual articles and the number of infoboxes in the source language. It is also limited by the quality of the translators. WikiCiKE attempts to mine the missing infoboxes directly from the article texts and thus achieves a higher recall compared with these methods as shown in Section 4.3.

### 5.3 Transfer Learning

Transfer learning can be grouped into four categories: instance-transfer, feature-representation-transfer, parameter-transfer and relational-knowledge-transfer (Pan and Yang, 2010). TrAdaBoost, the instance-transfer approach, is an extension of the AdaBoost algorithm, and demonstrates better transfer ability than tradition-

al learning techniques (Dai et al., 2007). Transfer learning have been widely studied for classification, regression, and cluster problems. However, few efforts have been spent in the information extraction tasks with knowledge transfer.

## 6 Conclusion and Future Work

In this paper we proposed a general cross-lingual knowledge extraction framework called WikiCiKE, in which extraction performance in the target Wikipedia is improved by using rich infoboxes in the source language. The problems of topic drift and translation error were handled by using the TrAdaBoost model. Chinese-English experimental results on four typical attributes showed that WikiCiKE significantly outperforms both the current translation based methods and the monolingual extraction methods. In theory, WikiCiKE can be applied to any two wiki knowledge based of different languages.

We have been considering some future work. Firstly, more attributes in more infobox templates should be explored to make our results much stronger. Secondly, knowledge in a minor language may also help improve extraction performance for a major language due to the cultural and religion differences. A bidirectional cross-lingual extraction approach will also be studied. Last but not least, we will try to extract multiple *attr-value* pairs at the same time for each article.

Furthermore, our work is part of a more ambitious agenda on exploitation of linked data. On the one hand, being able to extract data and knowledge from multilingual sources such as Wikipedia could help improve the coverage of linked data for applications. On the other hand, we are also investigating how to possibly integrate information, including subjective information (Sensoy et al., 2013), from multiple sources, so as to better support data exploitation in context dependent applications.

## Acknowledgement

The work is supported by NSFC (No. 61035004), NSFC-ANR (No. 61261130588), 863 High Technology Program (2011AA01A207), FP7-288342, FP7 K-Drive project (286348), the EPSRC WhatIf project (EP/J014354/1) and THU-NUS NExT Co-Lab. Besides, we gratefully acknowledge the assistance of Haixun Wang (MSRA) for improving the paper work.

## References

- S. Fissaha Adafre and M. de Rijke. 2006. Finding Similar Sentences across Multiple Languages in Wikipedia. *EACL 2006 Workshop on New Text: Wikis and Blogs and Other Dynamic Text Sources*.
- Sisay Fissaha Adafre and Maarten de Rijke. 2005. Discovering Missing Links in Wikipedia. *Proceedings of the 3rd International Workshop on Link Discovery*.
- Eytan Adar, Michael Skinner and Daniel S. Weld. 2009. Information Arbitrage across Multi-lingual Wikipedia. *WSDM'09*.
- David Aumüller, Hong Hai Do, Sabine Massmann and Erhard Rahm". 2005. Schema and ontology matching with COMA++. *SIGMOD Conference'05*.
- Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak and Sebastian Hellmann. 2009. DBpedia - A crystallization Point for the Web of Data. *J. Web Sem.*
- Christian Bizer, Tom Heath, Kingsley Idehen and Tim Berners-Lee. 2008. Linked data on the web (LDOW2008). *WWW'08*.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim S-turge and Jamie Taylor. 2008. Freebase: a Collaboratively Created Graph Database for Structuring Human Knowledge. *SIGMOD'08*.
- Gosse Bouma, Geert Kloosterman, Jori Mur, Gertjan Van Noord, Lonneke Van Der Plas and Jorg Tiedemann. 2008. Question Answering with Joost at CLEF 2007. *Working Notes for the CLEF 2008 Workshop*.
- Gosse Bouma, Sergio Duarte and Zahurul Islam. 2009. Cross-lingual Alignment and Completion of Wikipedia Templates. *CLIAWS3 '09*.
- Wenyuan Dai, Qiang Yang, Gui-Rong Xue and Yong Yu. 2007. Boosting for Transfer Learning. *ICML'07*.
- Dietterich and Thomas G. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Comput.*
- Sergio Ferrández, Antonio Toral, íscar Ferrández, Antonio Ferrández and Rafael Muñoz. 2009. Exploiting Wikipedia and EuroWordNet to Solve Cross-Lingual Question Answering. *Inf. Sci.*
- Aidan Finn and Nicholas Kushmerick. 2004. Multi-level Boundary Classification for Information Extraction. *ECML*.
- Achille Fokoue, Felipe Meneguzzi, Murat Sensoy and Jeff Z. Pan. 2012. Querying Linked Ontological Data through Distributed Summarization. *Proc. of the 26th AAAI Conference on Artificial Intelligence (AAAI2012)*.

- Yoav Freund and Robert E. Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.*
- Norman Heino and Jeff Z. Pan. 2012. RDFS Reasoning on Massively Parallel Hardware. *Proc. of the 11th International Semantic Web Conference (ISWC2012)*.
- Aidan Hogan, Jeff Z. Pan, Axel Polleres and Yuan Ren. 2011. Scalable OWL 2 Reasoning for Linked Data. *Reasoning Web. Semantic Technologies for the Web of Data*.
- Andreas Hotho, Robert Jäschke, Christoph Schmitz and Gerd Stumme. 2006. Information Retrieval in Folksonomies: Search and Ranking. *ESWC'06*.
- John D. Lafferty, Andrew McCallum and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *ICML'01*.
- Alberto Lavelli, MaryElaine Califf, Fabio Ciravegna, Dayne Freitag, Claudio Giuliano, Nicholas Kushmerick, Lorenza Romano and Neil Ireson. 2008. Evaluation of Machine Learning-based Information Extraction Algorithms: Criticisms and Recommendations. *Language Resources and Evaluation*.
- Juanzi Li, Jie Tang, Yi Li and Qiong Luo. 2009. RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. *IEEE Trans. Knowl. Data Eng.*
- Xiao Ling, Gui-Rong Xue, Wenyuan Dai, Yun Jiang, Qiang Yang and Yong Yu. 2008. Can Chinese Web Pages be Classified with English Data Source?. *WWW'08*.
- Sheila A. McIlraith, Tran Cao Son and Honglei Zeng. 2001. Semantic Web Services. *IEEE Intelligent Systems*.
- Thanh Hoang Nguyen, Viviane Moreira, Huong Nguyen, Hoa Nguyen and Juliana Freire. 2011. Multilingual Schema Matching for Wikipedia Infoboxes. *CoRR*.
- Jeff Z. Pan and Edward Thomas. 2007. Approximating OWL-DL Ontologies. *22nd AAAI Conference on Artificial Intelligence (AAAI-07)*.
- Jeff Z. Pan and Ian Horrocks. 2007. RDFS(FA): Connecting RDF(S) and OWL DL. *IEEE Transaction on Knowledge and Data Engineering*. 19(2): 192 - 206.
- Jeff Z. Pan and Ian Horrocks. 2006. OWL-Eu: Adding Customised Datatypes into OWL. *Journal of Web Semantics*.
- Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.*
- Nick Roussopoulos, Stephen Kelley and Frédéric Vincent. 1995. Nearest Neighbor Queries. *SIGMOD Conference'95*.
- Murat Sensoy, Achille Fokoue, Jeff Z. Pan, Timothy Norman, Yuqing Tang, Nir Oren and Katia Sycara. 2013. Reasoning about Uncertain Information and Conflict Resolution through Trust Revision. *Proc. of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS2013)*.
- Fabian M. Suchanek, Gjergji Kasneci and Gerhard Weikum. 2007. Yago: a Core of Semantic Knowledge. *WWW'07*.
- Max Volkel, Markus Krotzsch, Denny Vrandečić, Heiko Haller and Rudi Studer. 2006. Semantic Wikipedia. *WWW'06*.
- Zhichun Wang, Juanzi Li, Zhigang Wang and Jie Tang. 2012. Cross-lingual Knowledge Linking across Wiki Knowledge Bases. *21st International World Wide Web Conference*.
- Daniel S. Weld, Fei Wu, Eytan Adar, Saleema Amer-shi, James Fogarty, Raphael Hoffmann, Kayur Patel and Michael Skinner. 2008. Intelligence in Wikipedia. *AAAI'08*.
- Fei Wu and Daniel S. Weld. 2007. Autonomously Semantifying Wikipedia. *CIKM'07*.
- Fei Wu and Daniel S. Weld. 2010. Open Information Extraction Using Wikipedia. *ACL'10*.
- Fei Wu, Raphael Hoffmann and Daniel S. Weld. 2008. Information Extraction from Wikipedia: Moving down the Long Tail. *KDD'08*.
- Wentao Wu, Hongsong Li, Haixun Wang and Kenny Qili Zhu. 2012. Probase: a Probabilistic Taxonomy for Text Understanding. *SIGMOD Conference'12*.
- Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead and Stephen Soderland. 2007. TextRunner: Open Information Extraction on the Web. *NAACL-Demonstrations'07*.
- Xinfeng Zhang, Xiaozhao Xu, Yiheng Cai and Yaowei Liu. 2009. A Weighted Hyper-Sphere SVM. *ICNC(3)'09*.

# Bridging Languages through Etymology: The case of cross language text categorization

Vivi Nastase and Carlo Strapparava

Human Language Technologies, Fondazione Bruno Kessler

Trento, Italy

{nastase, strappa}@fbk.eu

## Abstract

We propose the hypothesis that word etymology is useful for NLP applications as a bridge between languages. We support this hypothesis with experiments in cross-language (English-Italian) document categorization. In a straightforward bag-of-words experimental set-up we add etymological ancestors of the words in the documents, and investigate the performance of a model built on English data, on Italian test data (and viceversa). The results show not only statistically significant, but a large improvement – a jump of almost 40 points in F1-score – over the raw (vanilla bag-of-words) representation.

## 1 Introduction

When exposed to a document in a language he does not know, a reader might be able to glean some meaning from words that are the same (e.g. names) or similar to those in a language he knows. As an example, let us say that an Italian speaker is reading an English text that contains the word *expense*, which he does not know. He may be reminded however of the Latin word *expensa* which is also the etymological root of the Italian word *spesa*, which usually means “cost”/“shopping”, and may thus infer that the English word refers to the cost of things. In the experiments presented here we investigate whether an automatic text categorization system could benefit from knowledge about the etymological roots of words. The cross language text categorization (CLTC) task consists of categorizing documents in a target language  $L_t$  using a model built from labeled examples in a source language  $L_s$ . The task becomes more difficult when the data consists of comparable corpora in the two languages – documents on the same topics (e.g. sports, economy) – instead of parallel corpora – there exists a one-to-one correspondence

between documents in the corpora for the two languages, one document being the translation of the other.

To test the usefulness of etymological information we work with comparable collections of news articles in English and Italian, whose articles are assigned one of four categories: *culture\_and\_school*, *tourism*, *quality\_of\_life*, *made\_in\_Italy*. We perform a progression of experiments, which embed etymological information deeper and deeper into the model. We start with the basic set-up, representing the documents as bag-of-words, where we train a model on the English training data, and use this model to categorize documents from the Italian test data (and viceversa). The results are better than random, but quite low. We then add the etymological roots of the words in the data to the bag-of-words, and notice a large – 21 points – increase in performance in terms of F1-score. We then use the bag-of-words representation of the training data to build a semantic space using LSA, and use the generated word vectors to represent the training and test data. The improvement is an additional 16 points in F1-score.

Compared to related work, presented in Section 3, where cross language text categorization is approached through translation or mapping of features (i.e. words) from the source to the target language, word etymologies are a novel source of cross-lingual knowledge. Instead of mapping features between languages, we introduce new features which are shared, and thus do not need translation or other forms of mapping.

The experiments presented show unequivocally that word etymology is a useful addition to computational models, just as they are to readers who have such knowledge. This is an interesting and useful result, especially in the current research landscape where using and exploiting multi-linguality is a desired requirement.

morpheme	relation	related morpheme	
eng: ex-	rel:etymological_origin_of	eng: excentric	
eng: expense	rel:etymology	lat: expensa	
eng: -ly	rel:etymological_origin_of	eng: absurdly	English: <i>muscle</i>
eng: -ly	rel:etymological_origin_of	eng: admirably	↓
...			French: <i>muscle</i>
ita: spesa	rel:etymology	lat: expensa	↓
ita: spesa	rel:has_derived_form	ita: spese	Latin: <i>musculus</i>
...			↓
ita: spesare	rel:etymologically_related	ita: spesa	Latin: <i>mus</i>
...			↓
lat: expensa	rel:etymological_origin_of	eng: expense	Proto Indo-European: <i>muh<sub>2</sub>s</i>
lat: expensa	rel:etymological_origin_of	ita: spesa	
...			
lat: expensa	rel:is_derived_from	lat: expensus	
...			

Figure 1: Sample entries from the Etymological WordNet, and a few etymological layers

## 2 Word Etymology

Word etymology gives us a glimpse into the evolution of words in a language. Words may be adopted from a language because of cultural, scientific, economic, political or other reasons (Hitchings, 2009). In time these words “adjust” to the language that adopted them – their sense may change to various degrees – but they are still semantically related to their etymological roots. To illustrate the point, we show an example that the reader, too, may find amusing: on the ticket validation machine on Italian buses, by way of instruction, it is written *Per obliterare il biglietto ....* A native/frequent English speaker would most probably key in on, and be puzzled by, the word *obliterare*, very similar to the English *obliterate*, whose most used sense is *to destroy completely / cause to physically disappear*. The Italian *obliterare* has the “milder” sense of *cancellare* – *cancel* (which is also shared by the English *obliterate*, but is less frequent according to Merriam-Webster), and both come from the Latin *obliterare* – erase, efface, cause to disappear. While there has been some sense migration – in English the more (physically) destructive sense of the word has higher prominence, while in Italian the word is closer in meaning to its etymological root – the Italian and the English words are still semantically related.

Dictionaries customarily include etymologi-

cal information for their entries, and recently, Wikipedia’s Wiktionary has joined this trend. The etymological information can, and indeed has been extracted and prepared for machine consumption (de Melo and Weikum, 2010): Etymological WordNet<sup>1</sup> contains 6,031,431 entries for 2,877,036 words (actually, morphemes) in 397 languages. A few sample entries from this resource are shown in Figure 1.

The information in Etymological WordNet is organized around 5 relations: *etymology* with its inverse *etymological\_origin\_of*; *is\_derived\_from* with its inverse *has\_derived\_form*; and the symmetrical *etymologically\_related*. The *etymology* relation links a word with its etymological ancestors, and it is the relation used in the experiments presented here. Prefixes and suffixes – such as *ex-* and *-ly* shown in Figure 1 – are filtered out, as they bring in much noise by relating words that merely share such a morpheme (e.g. *absurdly* and *admirably*) but are otherwise semantically distant. *has\_derived\_form* is also used, to capture morphological variations.

The depth of the etymological hierarchy (considering the *etymology* relations) is 10. Figure 1 shows an example of a word with several levels of etymological ancestry.

<sup>1</sup><http://www1.icsi.berkeley.edu/~demelo/etymwn/>

		English texts					Italian texts				
		$t_1^e$	$t_2^e$	$\dots$	$t_{n-1}^e$	$t_n^e$	$t_1^i$	$t_2^i$	$\dots$	$t_{m-1}^i$	$t_m^i$
English Lexicon	$w_1^e$	0	1	$\dots$	0	1	0	0	$\dots$		
	$w_2^e$	1	1	$\dots$	1	0	0	$\dots$			
	$\vdots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\vdots$		0		$\vdots$
	$w_{p-1}^e$	0	1	$\dots$	0	0			$\dots$	$\dots$	0
	$w_p^e$	0	1	$\dots$	0	0			$\dots$	0	0
shared names and words	$w_1^{e/i}$	1	0	$\dots$	0	0	0	0	$\dots$	0	1
	$\vdots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
common etymology	$w_1^{ety}$	0	1	$\dots$	0	0	0	0	$\dots$	1	0
	$\vdots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
Italian Lexicon	$w_1^i$	0	0	$\dots$			0	1	$\dots$	1	1
	$w_2^i$	0	$\dots$				1	1	$\dots$	0	1
	$\vdots$	$\vdots$		0		$\vdots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
	$w_{q-1}^i$			$\dots$	$\dots$	0	0	1	$\dots$	0	1
	$w_q^i$			$\dots$	0	0	0	1	$\dots$	1	0

Figure 2: Multilingual word-by-document matrix

### 3 Cross Language Text Categorization

Text categorization (also text classification), “the task of automatically sorting a set of documents into categories (or classes or topics) from a predefined set” (Sebastiani, 2005), allows for the quick selection of documents from the same domain, or the same topic. It is a very well research area, dating back to the 60s (Borko and Bernick, 1962). The most frequently, and successfully, used document representation is the bag-of-words (BoWs). Results using this representation achieve accuracy in the 90%. Most variations include feature filtering or weighing, and variations in learning algorithms (Sebastiani, 2005).

Within the area of cross-language text categorization (CLTC) several methods have been explored for producing the model for a target language  $L_t$  using information and data from the source language  $L_s$ . In a precursor task to CLTC, cross language information retrieval (CLIR), Dumais et al. (1997) find semantic correspondences in parallel (different language) corpora through latent semantic analysis (LSA). Most CLTC methods rely heavily on machine translation (MT). MT has been used: to cast the cross-language text

categorization problem to the monolingual setting (Fortuna and Shawe-Taylor, 2005); to cast the cross-language text categorization problem into two monolingual settings for active learning (Liu et al., 2012); to translate and adapt a model built on language  $L_s$  to language  $L_t$  (Rigutini et al., 2005), (Shi et al., 2010); to produce parallel corpora for multi-view learning (Guo and Xiao, 2012). Wan et al. (2011) also use machine translation, but enhance the processing through domain adaptation by feature weighing, assuming that the training data in one language and the test data in the other come from different domains, or can exhibit different linguistic phenomena due to linguistic and cultural differences. Prettenhofer and Stein (2010) use a word translation oracle to produce *pivots* – pairs of semantically similar words – and use the data partitions induced by these words to find cross language structural correspondences.

In a computationally lighter framework, not dependent on MT, Gliozzo and Strapparava (2006) and Wu et al. (2008) use bilingual lexicons and aligned WordNet synsets to obtain shared features between the training data in language  $L_s$  and the testing data in language  $L_t$ . Gliozzo and Strapparava (2005), the first to use comparable as op-

posed to parallel corpora for CLTC, use LSA to build multilingual domain models.

The bag-of-words document representation maps a document  $d_i$  from a corpus  $D$  into a  $k$ -dimensional space  $\mathbb{R}^k$ , where  $k$  is the dimension of the (possibly filtered) vocabulary of the corpus:  $W = \{w_1, \dots, w_k\}$ . Position  $j$  in the vector representation of  $d_i$  corresponds to word  $w_j$ , and it may have different values, among the most commonly used being: binary values –  $w_j$  appears (1) or not (0) in  $d_i$ ; frequency of occurrence of  $w_j$  in  $d_i$ , absolute or normalized (relative to the size of the document or the size of the vocabulary); the  $tf * idf(w_j, d_i, D)$ .

For the task of cross language text categorization, the problem of sharing a model across languages is that the dimensions, a.k.a the vocabulary, of the two languages are largely different. Limited overlap can be achieved through shared names and words. As we have seen in the literature review, machine translation and bilingual dictionaries can be used to cast these dimensions from the source language  $L_s$  to the target language  $L_t$ . In this work we explore expanding the shared dimensions through word etymologies. Figure 2 shows schematically the binary  $k$  dimensional representation for English and Italian data, and shared dimensions.

Cross language text categorization could be used to obtain comparable corpora for building translation models. In such a situation, relying on a framework that itself relies on machine translation is not helpful. Bilingual lexicons are available for frequently studied languages, but less so for those poorer in resources. Considering such shortcomings, we look into additional linguistic information, in particular word etymology. This information impacts the data representation, by introducing new shared features between the different language corpora without the need for translation or other forms of mapping. The newly produced representation can be used in conjunction with any of the previously proposed algorithms.

Word etymologies are a novel source of linguistic information in NLP, possibly because resources that capture this information in a machine readable format are also novel. Fang et al. (2009) used limited etymological information extracted from the Collins English Dictionary (CED) for text categorization on the British National Corpus (BNC): information on the provenance of words (ranges of

probability distribution of etymologies in different versions of Latin – New Latin, Late Latin, Medieval Latin) was used in a “home-made” range classifier.

The experiments presented in this paper use the bag-of-words document representation with absolute frequency values. To this basic representation we add word etymological ancestors and run classification experiments. We then use LSA – previously shown by (Dumais et al., 1997) and (Gliozzo and Strapparava, 2005) to be useful for this task – to induce the latent semantic dimensions of documents and words respectively, hypothesizing that word etymological ancestors will lead to semantic dimensions that transcend language boundaries. The vectors obtained through LSA (on the training data only) for words that are shared by the English training data and the Italian test data (names, and most importantly, etymological ancestors of words in the original documents) are then used for re-representing the training and test data. The same process is applied for Italian training and English test data. Classification is done using support vector machines (SVMs).

### 3.1 Data

The data we work with consists of comparable corpora of news articles in English and Italian. Each news article is annotated with one of the four categories: *culture\_and\_school*, *tourism*, *quality\_of\_life*, *made\_in\_Italy*. Table 1 shows the dataset statistics. The average document length is approximately 300 words.

### 3.2 Raw cross-lingual text categorization

As is commonly done in text categorization (Sebastiani, 2005), the documents in our data are represented as bag-of-words, and classification is done using support vector machines (SVMs).

One experimental run consists of 4 binary experiments – one class versus the rest, for each of the 4 classes. The results are reported through micro-averaged precision, recall and F1-score for the targeted class, as well as overall accuracy. The high results, on a par with text categorization experiments in the field, validates our experimental set-up.

For the cross language categorization experiments described in this paper, we use the data described above, and train on one language (English/Italian), and test on the other, using the same

Categories	English			Italian		
	Training	Test	Total	Training	Test	Total
quality_of_life	5759	1989	7748	5781	1901	7682
made_in_Italy	5711	1864	7575	6111	2068	8179
tourism	5731	1857	7588	6090	2015	8105
culture_and_school	3665	1245	4910	6284	2104	8388
Total	20866	6955	27821	24266	8088	32354

Table 1: Dataset statistics

<i>monolingual BoW categorization</i>				
	Prec	Rec	F1	Acc
Train EN / Test EN	0.92	0.92	0.92	0.96
Train IT / Test IT	0.94	0.94	0.94	0.97

Table 2: Performance for monolingual raw text categorization

experimental set-up as for the monolingual scenario (4 binary problems). The categorization baseline (BoW\_baseline in Figure 4) was obtained in this set-up. This baseline is higher than the random baseline or the positive class baseline<sup>2</sup> (all instances are assigned the target class in each of the 4 binary classification experiments) due to shared words and names between the two languages.

### 3.3 Enriching the bag-of-word representation with word etymology

As personal experience has shown us that etymological information is useful for comprehending a text in a different language, we set out to test whether this information can be useful in an automatic processing setting. We first verified whether the vocabularies of our two corpora, English and Italian, have shared word etymologies. Relying on word etymologies from the Etymological dictionary, we found that from our data’s vocabulary, 518 English terms and 543 Italian terms shared 490 direct etymological ancestors. Etymological ancestors also help cluster related terms within one language – 887 etymological ancestors for 4727 English and 864 ancestors for 5167 Italian terms. This overlap further increases when adding derived forms (through the *has\_derived\_form* relation). The fact that this overlap exists strengthens the motivation to try using etymological ancestors for the task of text categorization.

In this first step of integrating word etymology

<sup>2</sup>In this situation the random and positive class baseline are the same: 25% F1 score.

into the experiment, we extract for each word in each document in the dataset its ancestors from the Etymological dictionary. Because each word  $w_j$  in a document  $d_i$  has associated an absolute frequency value  $f_{ij}$  (the number of occurrences of  $w_j$  in  $d_i$ ), for the added etymological ancestors  $e_k$  in document  $D_i$  we associate as value the sum of frequencies of their etymological children in  $d_i$ :

$$f_{ie_k} = \sum_{\substack{w_j \in d_i \\ w_j \text{ etymology } e_k}} f_{ij}$$

We make the depth of extraction a parameter, and generate data representation when considering only direct etymological antecedents (depth 1) and then up to a distance of N. For our dataset we noticed that the representation does not change after N=4, so this is the maximum depth we consider. The bag-of-words representation for each document is expanded with the corresponding etymological features.

expansion	training data vocabulary size	vocabulary overlap with testing
Train EN /Test IT		
raw	71122	14207 (19.9%)
depth 1	78936	18275 (23.1%)
depth 2	79068	18359 (23.2%)
depth 3	79100	18380 (23.2%)
depth 4	79103	18382 (23.2%)
Train IT /Test EN		
raw	78750	14110 (17.9%)
depth 1	83656	18682 (22.3%)
depth 2	83746	18785 (22.4%)
depth 3	83769	18812 (22.5%)
depth 4	83771	18814 (22.5%)

Table 3: Feature expansion with word etymologies

Table 3 shows the training data vocabulary size and increase in the overlap between the training and test data with the addition of etymological fea-

tures. The increase is largest when introducing the immediate etymological ancestors, of approximately 4000 new (overlapping) features for both combinations of training and testing. Without etymological features the overlap was approximately 14000 for both configurations. The results obtained with this enriched BoW representation for etymological ancestor depth 1, 2 and 3 are presented in Figure 4.

### 3.4 Cross-lingual text categorization in a latent semantic space adding etymology

Shared word etymologies can serve as a bridge between two languages as we have seen in the previous configuration. When using shared word etymologies in the bag-of-words representation, we only take advantage of the shallow association between these new features and the classes within which they appear. But through the co-occurrence of the etymological features and other words in different documents in the training data, we can induce a deeper representation for the words in a document, that captures better the relationship between the features (words) and the classes to which the documents belong. We use latent semantic analysis (LSA) (Deerwester et al., 1990) to perform this representational transformation. The process relies on the assumption that word co-occurrences across different documents are the surface manifestation of shared semantic dimensions. Mathematically, the  $\langle \text{word} \times \text{document} \rangle$  matrix  $D$  is expressed as a product of three matrices:

$$D = V\Sigma U^T$$

by performing singular value decomposition (SVD).  $V$  would correspond roughly to a  $\langle \text{word} \times \text{latent semantic dimension} \rangle$  matrix,  $U^T$  is the transposed of a  $\langle \text{document} \times \text{latent semantic dimension} \rangle$  matrix, and  $\Sigma$  is a diagonal matrix whose values are indicative of the “strength” of the semantic dimensions. By reducing the size of  $\Sigma$ , for example by selecting the dimensions with the top  $K$  values, we can obtain an approximation of the original matrix  $D \approx D_K = V_K \Sigma_K U_K^T$ , where we restrict the latent semantic dimensions taken into account to the  $K$  chosen ones. Figure 3 shows schematically the process.

We perform this decomposition and dimension reduction step on the  $\langle \text{word} \times \text{document} \rangle$  matrix built from the training data only, and using  $K=400$ . Both the training and test data are then

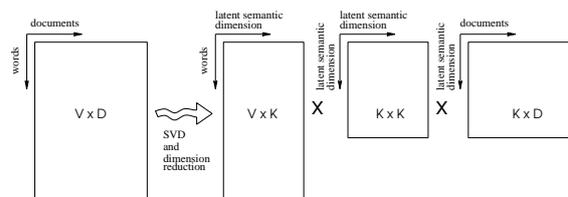


Figure 3: Schematic view of LSA

re-represented through the new word vectors from matrix  $V_K$ . Because the LSA space was built only from the training data, only the shared words and shared etymological ancestors are used to produce representations of the test data. The categorization is done again with SVM. The results of this experiment are shown in Figure 4, together with an LSA baseline – using the raw data and relying on shared words and names as overlap.

## 4 Discussion

The experiments whose results we present here were produced using unfiltered data – all words in the datasets, all etymological ancestors up to the desired depth, no filtering based on frequency of occurrence. Feature filtering is commonly done in machine learning when the data has many features, and in text categorization when using the bag-of-words representation in particular. We chose not to perform this step for two main reasons: (i) filtering is sensitive to the chosen threshold; (ii) LSA thrives on word co-occurrences, which would be drastically reduced by word removal. The point that etymology information is a useful addition to the task of cross-language text categorization can be made without finding the optimal filtering set-up.

The baseline experiments show that despite the relatively large word overlap (approx. 14000 terms), cross-language text categorization gives low results. Adding a first batch of etymological information – approximately 4000 shared immediate ancestors – leads to an increase of 18 points in terms of F1-score on the BoW experimental set-up for English training/Italian testing, and 21 points for Italian training/English testing. Further additions of etymological ancestors at depths 2 and 3 results in an increase of 21 points in terms of F1-score for English training/Italian testing, and 27 points for Italian training/English testing. The higher increase in performance on this experimental configuration for Italian training/English testing is explained by the higher term overlap be-

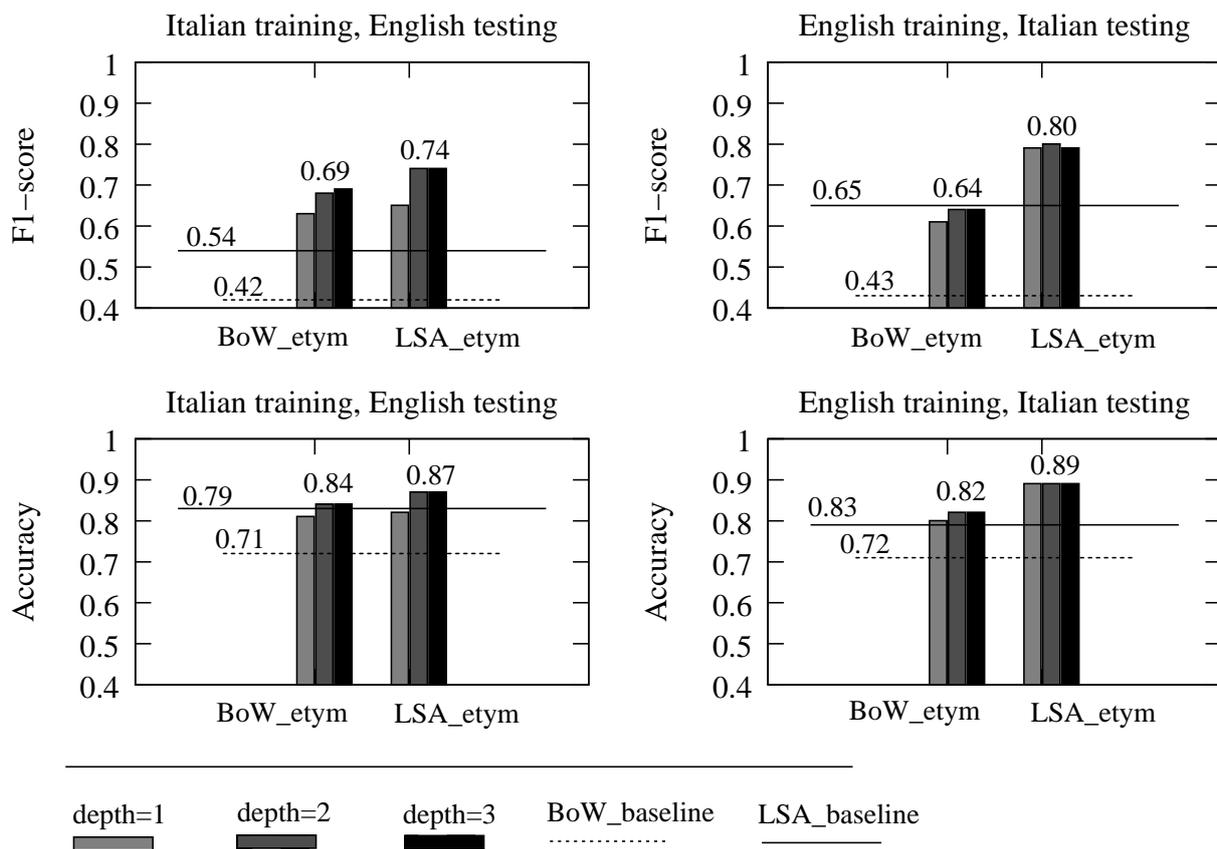


Figure 4: CLTC results with etymological features

tween the training and test data, as evidenced by the statistics in Table 3.

The next processing step induced a representation of the shared words that encodes deeper level dependencies between words and documents based on word co-occurrences in documents. The LSA space built on the training data leads to a vector representation of the shared words, including the shared etymological ancestors, that captures more than the obvious word-document co-occurrences. Using this representation leads to a further increase of 15 points in F1-score for English training/Italian testing set-up over the BoW representation, and 14 points over the baseline LSA-based categorization. The increase for the Italian training/English testing is 5 points over the BoW representation, but 20 points over the baseline LSA. We saw that the high performance BoW on Italian training/English testing is due to the high term overlap. The clue to why the increase when using LSA is lower than for English training/Italian testing is in the way LSA operates – it relies heavily on word co-occurrences in finding the latent semantic dimensions of documents and words. We expect then that in the Italian training

collection, words are “less shared” among documents, which means a lower average document frequency. Figure 5 shows the changes in average document frequency for the two training collections, starting with the raw data (depth 0), and with additional etymological features.

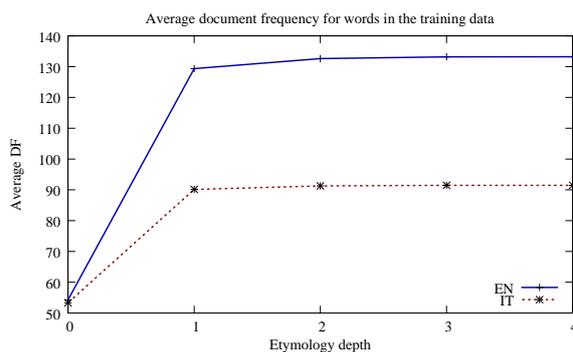


Figure 5: Document frequency changes with the addition of etymological features

The shape of the document frequency curves mirror the LSA results – the largest increase is the effect of adding the set of direct etymological ancestors, and additions of further, more distant, ancestors lead to smaller improvements.

We have performed the experiments described above on two releases of the Etymological dictionary. The results described in the paper were obtained on the latest release (February 2013). The difference in results on the two dictionary versions was significant: a 4 and 5 points increase respectively in micro-averaged F1-score in the bag-of-words setting for English training/Italian testing and Italian training/English testing, and a 2 and 6 points increase in the LSA setting. This indicates that more etymological information is better, and the dynamic nature of Wikipedia and the Wiktionary could lead to an ever increasing and better etymological resource for NLP applications.

## 5 Conclusion

The motivation for this work was to test the hypothesis that information about word etymology is useful for computational approaches to language, in particular for text classification. Cross-language text classification can be used to build comparable corpora in different languages, using a single language starting point, preferably one with more resources, that can thus spill over to other languages. The experiments presented have shown clearly that etymological ancestors can be used to provide the necessary bridge between the languages we considered – English and Italian. Models produced on English data when using etymological information perform with high accuracy (89%) and high F1-score (80) on Italian test data, with an increase of almost 40 points over a simple bag-of-words model, which, for crossing language boundaries, relies exclusively on shared names and words. Training on Italian data and testing on English data performed almost as well (87% accuracy, 75 F1-score). We plan to expand our experiments to more languages with shared etymologies, and investigate what characteristics of languages and data indicate that etymological information is beneficial for the task at hand.

We also plan to explore further uses for this language bridge, at a finer semantic level. Monolingual and cross-lingual textual entailment in particular would be interesting applications, because they require finding shared meaning on two text fragments. Word etymologies would allow recognizing words with shared ancestors, and thus with shared meaning, both within and across languages.

## Acknowledgements

We thank the reviewers for the helpful comments. This work was financially supported by the EC-funded project EXCITEMENT – EXploring Customer Interactions through Textual EntailMENT FP7 ICT-287923. Carlo Strapparava was partially supported by the PerTe project (Trento RISE).

## References

- Harold Borko and Myrna Bernick. 1962. *Automatic Document Classification*. System Development Corporation, Santa Monica, CA.
- Gerard de Melo and Gerhard Weikum. 2010. Towards universal multilingual knowledge bases. In *Principles, Construction, and Applications of Multilingual Wordnets. Proceedings of the 5th Global WordNet Conference (GWC 2010)*, pages 149–156, New Delhi, India.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.
- Susan T. Dumais, Todd A. Letsche, Michael L. Littman, and Thomas K. Landauer. 1997. Automatic cross-language retrieval using latent semantic indexing. In *AAAI Symposium on CrossLanguage Text and Speech Retrieval*.
- Alex Chengyu Fang, Wanyin Li, and Nancy Ide. 2009. Latin etymologies as features on BNC text categorization. In *23rd Pacific Asia Conference on Language, Information and Computation (PACLIC 2009)*, pages 662–669.
- Blaz Fortuna and John Shawe-Taylor. 2005. The use of machine translation tools for cross-lingual text mining. In *Learning with multiple views – Workshop at the 22nd International Conference on Machine Learning (ICML 2005)*.
- Alfio Gliozzo and Carlo Strapparava. 2005. Cross language text categorization by acquiring multilingual domain models from comparable corpora. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*.
- Alfio Gliozzo and Carlo Strapparava. 2006. Exploiting comparable corpora and bilingual dictionaries for cross-language text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 553–560, Sydney, Australia.

- Yuhong Guo and Min Xiao. 2012. Cross language text classification via subspace co-regularized multi-view learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, Edinburgh, Scotland, UK.
- Henry Hitchings. 2009. *The Secret Life of Words: How English Became English*. John Murray Publishers.
- Yue Liu, Lin Dai, Weitao Zhou, and Heyan Huang. 2012. Active learning for cross language text categorization. In *Proceedings of the 16th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining (PAKDD 2012)*, pages 195–206, Kuala Lumpur, Malaysia.
- Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 1118–1127, Uppsala, Sweden.
- Leonardo Rigutini, Marco Maggini, and Bing Liu. 2005. An EM based training algorithm for cross-language text categorization. In *Proceedings of the International Conference on Web Intelligence (WI 2005)*, pages 200–206, Compiegne, France.
- Fabrizio Sebastiani. 2005. Text categorization. In Alessandro Zanasi, editor, *Text Mining and its Applications*, pages 109–129. WIT Press, Southampton, UK.
- Lei Shi, Rada Mihalcea, and Minhjun Tian. 2010. Cross language text classification by model translation and semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 1057–1067, Uppsala, Sweden.
- Chang Wan, Rong Pan, and Jifei Li. 2011. Bi-weighting domain adaptation for cross-language text classification. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*, pages 1535–1540, Barcelona, Catalonia, Spain.
- Ke Wu, Xiaolin Wang, and Bao-Liang Lu. 2008. Cross language text categorization using a bilingual lexicon. In *Third International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 165–172, Hyderabad, India.

# Creating Similarity: Lateral Thinking for Vertical Similarity Judgments

**Tony Veale**

Web Science and Technology Division,  
Korean Advanced Institute of Science  
and Technology, Yuseong, South Korea  
Tony.Veale@gmail.com

**Guofu Li**

School of Computer Science and Informatics,  
University College Dublin,  
Belfield, Dublin D2, Ireland.  
li.guofu.l@gmail.com

## Abstract

Just as *observing* is more than just *seeing*, *comparing* is far more than mere *matching*. It takes understanding, and even inventiveness, to discern a useful basis for judging two ideas as similar in a particular context, especially when our perspective is shaped by an act of linguistic creativity such as metaphor, simile or analogy. Structured resources such as WordNet offer a convenient hierarchical means for converging on a common ground for comparison, but offer little support for the divergent thinking that is needed to creatively view one concept as another. We describe such a means here, by showing how the web can be used to harvest many divergent views for many familiar ideas. These lateral views complement the vertical views of WordNet, and support a system for idea exploration called *Thesaurus Rex*. We show also how *Thesaurus Rex* supports a novel, generative similarity measure for WordNet.

## 1 Seeing is Believing (*and* Creating)

Similarity is a cognitive phenomenon that is both complex and subjective, yet for practical reasons it is often modeled as if it were simple and objective. This makes sense for the many situations where we want to align our similarity judgments with those of others, and thus focus on the same conventional properties that others are also likely to focus upon. This reliance on the consensus viewpoint explains why WordNet (Fellbaum, 1998) has proven so useful as a basis for computational measures of lexico-semantic similarity

(e.g. see Pederson *et al.* 2004, Budanitsky & Hirst, 2006; Seco *et al.* 2006). These measures reduce the similarity of two lexical concepts to a single number, by viewing similarity as an objective estimate of the overlap in their salient qualities. This convenient perspective is poorly suited to creative or insightful comparisons, but it is sufficient for the many mundane comparisons we often perform in daily life, such as when we organize books or look for items in a supermarket. So if we do not know in which aisle to locate a given item (such as *oatmeal*), we may tacitly know how to locate a similar product (such as *cornflakes*) and orient ourselves accordingly.

Yet there are occasions when the recognition of similarities spurs the *creation* of similarities, when the act of comparison spurs us to invent new ways of looking at an idea. By placing *pop tarts* in the breakfast aisle, food manufacturers encourage us to view them as a breakfast food that is not dissimilar to *oatmeal* or *cornflakes*. When ex-PM Tony Blair published his memoirs, a mischievous activist encouraged others to move his book from *Biography* to *Fiction* in bookshops, in the hope that buyers would see it in a new light. Whenever we use a novel metaphor to convey a non-obvious viewpoint on a topic, such as “*cigarettes are time bombs*”, the comparison may spur us to insight, to see aspects of the topic that *make* it more similar to the vehicle (see Ortony, 1979; Veale & Hao, 2007).

In formal terms, assume agent A has an insight about concept X, and uses the metaphor *X is a Y* to also provoke this insight in agent B. To arrive at this insight for itself, B must intuit what X and Y have in common. But this commonality is surely more than a standard categorization of X, or else it would not count as an insight about X. To understand the metaphor, B must place X

in a new category, so that X can be seen as more similar to Y. Metaphors shape the way we perceive the world by re-shaping the way we make similarity judgments. So if we want to imbue computers with the ability to make and to understand creative metaphors, we must first give them the ability to look beyond the narrow viewpoints of conventional resources.

Any measure that models similarity as an objective function of a conventional worldview employs a *convergent* thought process. Using WordNet, for instance, a similarity measure can vertically converge on a common superordinate category of both inputs, and generate a single numeric result based on their distance to, and the information content of, this common generalization. So to find the most conventional ways of seeing a lexical concept, one simply ascends a narrowing concept hierarchy, using a process de Bono (1970) calls *vertical thinking*. To find novel, non-obvious and useful ways of looking at a lexical concept, one must use what Guilford (1967) calls *divergent thinking* and what de Bono calls *lateral thinking*. These processes cut across familiar category boundaries, to simultaneously place a concept in many different categories so that we can see it in many different ways.

de Bono argues that vertical thinking is selective while lateral thinking is generative. Whereas vertical thinking concerns itself with the “right” way or a single “best” way of looking at things, lateral thinking focuses on producing alternatives to the status quo. To be as useful for creative tasks as they are for conventional tasks, we need to re-imagine our computational similarity measures as generative rather than selective, expansive rather than reductive, divergent as well as convergent and lateral as well as vertical. Though WordNet is ideally structured to support vertical, convergent reasoning, its comprehensive nature means it can also be used as a solid foundation for building a more lateral and divergent model of similarity. Here we will use the web as a source of diverse perspectives on familiar ideas, to complement the conventional and often narrow views codified by WordNet.

Section 2 provides a brief overview of past work in the area of similarity measurement, before section 3 describes a simple bootstrapping loop for acquiring richly diverse perspectives from the web for a wide variety of familiar ideas. These perspectives are used to enhance a WordNet-based measure of lexico-semantic similarity in section 4, by broadening the range of informative viewpoints the measure can select from.

Similarity is thus modeled as a process that is both generative *and* selective. This lateral-and-vertical approach is evaluated in section 5, on the Miller & Charles (1991) data-set. A web app for the lateral exploration of diverse viewpoints, named *Thesaurus Rex*, is also presented, before closing remarks are offered in section 6.

## 2 Related Work and Ideas

WordNet’s taxonomic organization of noun-senses and verb-senses – in which very general categories are successively divided into increasingly informative sub-categories or instance-level ideas – allows us to gauge the overlap in information content, and thus of meaning, of two lexical concepts. We need only identify the deepest point in the taxonomy at which this content starts to diverge. This point of divergence is often called the LCS, or *least common subsumer*, of two concepts (Pederson *et al.*, 2004). Since sub-categories add new properties to those they inherit from their parents – Aristotle called these properties the *differentia* that stop a category system from trivially collapsing into itself – the depth of a lexical concept in a taxonomy is an intuitive proxy for its information content. Wu & Palmer (1994) use the depth of a lexical concept in the WordNet hierarchy as such a proxy, and thereby estimate the similarity of two lexical concepts as twice the depth of their LCS divided by the sum of their individual depths.

Leacock and Chodorow (1998) instead use the length of the shortest path between two concepts as a proxy for the conceptual distance between them. To connect any two ideas in a hierarchical system, one must vertically ascend the hierarchy from one concept, change direction at a potential LCS, and then descend the hierarchy to reach the second concept. (Aristotle was also first to suggest this approach in his *Poetics*). Leacock and Chodorow normalize the length of this path by dividing its size (in nodes) by twice the depth of the deepest concept in the hierarchy; the latter is an upper bound on the distance between any two concepts in the hierarchy. Negating the log of this normalized length yields a corresponding similarity score. While the role of an LCS is merely implied in Leacock and Chodorow’s use of a *shortest path*, the LCS is pivotal nonetheless, and like that of Wu & Palmer, the approach uses an essentially vertical reasoning process to identify a single “best” generalization.

Depth is a convenient proxy for information content, but more nuanced proxies can yield

more rounded similarity measures. Resnick (1995) draws on information theory to define the information content of a lexical concept as the negative log likelihood of its occurrence in a corpus, either explicitly (via a direct mention) or by presupposition (via a mention of any of its sub-categories or instances). Since the likelihood of a general category occurring in a corpus is higher than that of any of its sub-categories or instances, such categories are more predictable, and less informative, than rarer categories whose occurrences are less predictable and thus more informative. The negative log likelihood of the most informative LCS of two lexical concepts offers a reliable estimate of the amount of information shared by those concepts, and thus a good estimate of their similarity. Lin (1998) combines the intuitions behind Resnick's metric and that of Wu and Palmer to estimate the similarity of two lexical concepts as an information ratio: twice the information content of their LCS divided by the sum of their individual information contents.

Jiang and Conrath (1997) consider the converse notion of *dissimilarity*, noting that two lexical concepts are dissimilar to the extent that each contains information that is not shared by the other. So if the information content of their most informative LCS is a good measure of what they *do* share, then the sum of their individual information contents, minus twice the content of their most informative LCS, is a reliable estimate of their dissimilarity.

Seco *et al.* (2006) presents a minor innovation, showing how Resnick's notion of information content can be calculated without the use of an external corpus. Rather, when using Resnick's metric (or that of Lin, or Jiang and Conrath) for measuring the similarity of lexical concepts in WordNet, one can use the category structure of WordNet itself to estimate information content. Typically, the more general a concept, the more descendants it will possess. Seco *et al.* thus estimate the information content of a lexical concept as the log of the sum of all its unique descendants (both direct and indirect), divided by the log of the total number of concepts in the entire hierarchy. Not only is this *intrinsic* view of information content convenient to use, without recourse to an external corpus, Seco *et al.* show that it offers a better estimate of information content than its extrinsic, corpus-based alternatives, as measured relative to average human similarity ratings for the 30 word-pairs in the Miller & Charles (1991) test set.

A similarity measure can draw on other

sources of information besides WordNet's category structures. One might eke out additional information from WordNet's textual glosses, as in Lesk (1986), or use category structures other than those offered by WordNet. Looking beyond WordNet, entries in the online encyclopedia Wikipedia are not only connected by a dense topology of lateral links, they are also organized by a rich hierarchy of overlapping categories. Strube and Ponzetto (2006) show how Wikipedia can support a measure of similarity (and relatedness) that better approximates human judgments than many WordNet-based measures. Nonetheless, WordNet can be a valuable component of a hybrid measure, and Agirre *et al.* (2009) use an SVM (support vector machine) to combine information from WordNet with information harvested from the web. Their best similarity measure achieves a remarkable **0.93** correlation with human judgments on the Miller & Charles word-pair set.

Similarity is not always applied to pairs of concepts; it is sometimes analogically applied to pairs of *pairs* of concepts, as in proportional analogies of the form *A is to B as C is to D* (e.g., *hacks are to writers as mercenaries are to soldiers*, or *chisels are to sculptors as scalpels are to surgeons*). In such analogies, one is really assessing the similarity of the unstated relationship between each pair of concepts: thus, mercenaries are soldiers whose allegiance is paid for, much as hacks are writers with income-driven loyalties; sculptors use chisels to carve stone, while surgeons use scalpels to cut or carve flesh. Veale (2004) used WordNet to assess the similarity of  $A:B$  to  $C:D$  as a function of the combined similarity of  $A$  to  $C$  and of  $B$  to  $D$ . In contrast, Turney (2005) used the web to pursue a more divergent course, to represent the tacit relationships of  $A$  to  $B$  and of  $C$  to  $D$  as points in a high-dimensional space. The dimensions of this space initially correspond to linking phrases on the web, before these dimensions are significantly reduced using *singular value decomposition*.

In the infamous SAT test, an analogy  $A:B::C:D$  has four other pairs of concepts that serve as likely distractors (e.g. *singer:songwriter* for *hack:writer*) and the goal is to choose the most appropriate  $C:D$  pair for a given  $A:B$  pairing. Using variants of Wu and Palmer (1994) on the 374 SAT analogies of Turney (2005), Veale (2004) reports a success rate of 38–44% using only WordNet-based similarity. In contrast, Turney (2005) reports up to 55% success on the same analogies, partly because his approach aims

to match implicit relations rather than explicit concepts, and in part because it uses a divergent process to gather from the web as rich a perspective as it can on these latent relationships.

## 2.1 Clever Comparisons Create Similarity

Each of these approaches to similarity is a *user* of information, rather than a *creator*, and each fails to capture how a creative comparison (such as a metaphor) can spur a listener to view a topic from an atypical perspective. Camac & Glucksberg (1984) provide experimental evidence for the claim that “metaphors do not use preexisting associations to achieve their effects [...] people use metaphors to create new relations between concepts.” They also offer a salutary reminder of an often overlooked fact: every comparison exploits information, but each is also a source of new information in its own right. Thus, “this cola is acid” reveals a different perspective on *cola* (e.g. as a *corrosive substance* or an *irritating food*) than “this acid is cola” highlights for *acid* (such as e.g., a *familiar substance*)

Veale & Keane (1994) model the role of similarity in realizing the long-term perlocutionary effect of an informative comparison. For example, to compare surgeons to butchers is to encourage one to see all surgeons as more *bloody*, *crude* or *careless*. The reverse comparison, of butchers to surgeons, encourages one to see butchers as more *skilled* and *precise*. Veale & Keane present a network model of memory, called *Sapper*, in which activation can spread between related concepts, thus allowing one concept to prime the properties of a neighbor. To interpret an analogy, *Sapper* lays down new activation-carrying bridges in memory between analogical counterparts, such as between *surgeon* & *butcher*, *flesh* & *meat*, and *scalpel* & *cleaver*. Comparisons can thus have lasting effects on how *Sapper* sees the world, changing the pattern of activation that arises when it primes a concept.

Veale (2003) adopts a similarly dynamic view of similarity in WordNet, showing how an analogical comparison can result in the automatic addition of new categories and relations to WordNet itself. Veale considers the problem of finding an analogical mapping between different parts of WordNet’s noun-sense hierarchy, such as between instances of *Greek god* and *Norse god*, or between the letters of different alphabets, such as of Greek and Hebrew. But no structural similarity measure for WordNet exhibits enough discernment to e.g. assign a higher similarity to

*Zeus* & *Odin* (each is the supreme deity of its pantheon) than to a pairing of *Zeus* and any other *Norse god*, just as no structural measure will assign a higher similarity to *Alpha* & *Aleph* or to *Beta* & *Beth* than to any random letter pairing.

A fine-grained category hierarchy permits fine-grained similarity judgments, and though WordNet is useful, its sense hierarchies are not especially fine-grained. However, we can automatically make WordNet subtler and more discerning, by adding new fine-grained categories to unite lexical concepts whose similarity is not reflected by any existing categories. Veale (2003) shows how a property that is found in the glosses of two lexical concepts, of the same depth, can be combined with their LCS to yield a new fine-grained parent category, so e.g. “supreme” + *deity* = *Supreme-deity* (for *Odin*, *Zeus*, *Jupiter*, etc.) and “1st” + *letter* = *1st-letter* (for *Alpha*, *Aleph*, etc.) Selected aspects of the textual similarity of two WordNet glosses – the key to similarity in Lesk (1986) – can thus be reified into an explicitly categorical WordNet form.

## 3 Divergent (Re)Categorization

To tap into a richer source of concept properties than WordNet’s glosses, we can use web n-grams. Consider these descriptions of a *cowboy* from the Google n-grams (Brants & Franz, 2006). The numbers to the right are Google frequency counts.

a	<i>lonesome</i>	cowboy	432
a	<i>mounted</i>	cowboy	122
a	<i>grizzled</i>	cowboy	74
a	<i>swaggering</i>	cowboy	68

To find the stable properties that can underpin a meaningful fine-grained category for *cowboy*, we must seek out the properties that are so often presupposed to be salient of all cowboys that one can use them to anchor a simile, such as “*swaggering like a cowboy*” or “*as grizzled as a cowboy*”. So for each property *P* suggested by Google n-grams for a lexical concept *C*, we generate a *like*-simile for verbal behaviors such as *swaggering* and an *as-as*-simile for adjectives such as *lonesome*. Each is then dispatched to Google as a phrasal query. We value quality over size, as these similes will later be used to find diverse viewpoints on the web via bootstrapping. We thus manually filter each web simile, to weed out any that are ill-formed, and those intended to be seen as ironic by their authors. This gives us a body of 12,000+ valid web similes.

Veale (2011, 2012, 2013) notes that web uses of the pattern “as *P* as *C*” are rife with irony. In contrast, web instances of “*P S* such as *C*” – where *S* denotes a superordinate of *C* – are rarely ironic. Hao & Veale (2010) exploit this fact to filter ironic comparisons from web similes, by re-expressing each “as *P* as *C*” simile as “*P* \* such as *C*” (using a wildcard \* to match any values for *S*) and looking for attested uses of this new form on the web. Since each hit will also yield a value for *S* via the wildcard \*, and a fine-grained category *P-S* for *C*, we use this approach here to harvest fine-grained categories from the web from most of our similes.

Once *C* is seen to be an exemplary member of the category *P-S*, such as *cola* in *fizzy-drink*, a targeted web search is used to find other members of *P-S*, via the anchored query “*P S* such as \* and *C*”. For example, “*fizzy drinks* such as \* and *cola*” will retrieve web texts in which \* is matched to *soda* or *lemonade*. Each new member can then be used to instantiate a further query, as in “*fizzy drinks* such as \* and *soda*”, to retrieve other members of *P-S*, such as *champagne* and *root beer*. This bootstrapping process runs in successive cycles, using doubly-anchored patterns that – following Kozareva *et al.* (2008) and Veale *et al.* (2009) – explicitly mention both the category to be populated (*P-S*) and a recently acquired member of this category (*C*).

As cautioned by Kozareva *et al.*, it is reckless to bootstrap from members to categories to members again if each enfilade of queries is likely to return noisy results. A reliable filter must be applied at each stage, to ensure that any member *C* that is placed in a category *P-S* is a sensible member of the category *S*. Only by filtering in this way can we stop the rapid accumulation of noise. For instance, a WordNet-based filter discards any categorization “*P S* such as *X* and *C*” where *X* does not denote a WordNet entry for which *S* does not denote a valid hypernym. Such a filter offers no creative latitude, however, since it forces every pairing of *C* and *P-S* to precisely obey WordNet’s category hierarchy. We thus use instead the *near-miss* filter described in Veale *et al.* (2009), in which *X* must denote a descendant of some direct hypernym of some sense of *S*. The filter does not (and cannot) determine whether *P* is salient for *X*. It merely assumes that if *P* is salient for *C*, it is salient for *X*.

Five successive cycles of bootstrapping are performed, using the 12,000+ web similes as a starting point. Consider *cola*: after 1 cycle, we acquire 14 new categories, such as *effervescent-*

*beverage* and *sweet-beverage*. After 2 cycles we acquire 43 categories; after 3 cycles, 72; after 4 cycles, 93; and after 5 cycles, we acquire 102 fine-grained perspectives on *cola*, such as *stimulating-drink* and *corrosive-substance*.

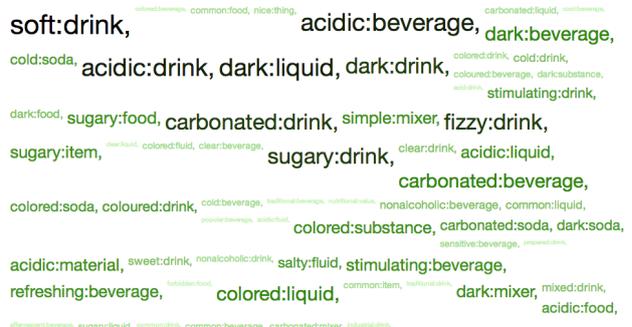


Figure 1. *Fine-grained perspectives for cola found by Thesaurus Rex on the web. See also Figures 3 and 4.*

These alternative viewpoints, for a broad array of concepts, are gleaned from the collective intelligence of the web. Some are more discerning and informative than others – see for instance *war & divorce* in Figure 4 – though as de Bono (1971) notes, lateral thinking does not privilege a narrow set of “correct” viewpoints, rather it generates a broad array of interesting alternatives, none of which are ever “wrong”, even if some prove more useful than others in a given context.

#### 4 Measuring and Creating Similarity

Which perspectives will be most useful and informative to a WordNet-based similarity metric? Simply, a perspective *M-C<sub>x</sub>* for a concept *C<sub>y</sub>* can be coherently added to WordNet *iff* *C<sub>x</sub>* denotes a hypernym of some sense of *C<sub>y</sub>* in WordNet. For purposes of quantifying the similarity of two terms *t<sub>1</sub>* and *t<sub>2</sub>* – by finding the WordNet senses of these terms that exhibit the highest similarity – we can augment WordNet with the perspectives on *t<sub>1</sub>* and *t<sub>2</sub>* that are coherent with WordNet’s hierarchy. So for *t<sub>1</sub>=cola* & *t<sub>2</sub>=acid, corrosive-substance* offers a coherent new perspective on each, slotting in beneath the matching WordNet sense of *substance*.

A category system is a structured feature space. We estimate the similarity of *C<sub>1</sub>* and *C<sub>2</sub>* as the cosine of the angle between the feature vectors that are constructed for each. The dimensions of these vectors are the atomic hypernyms (direct or indirect) of *C<sub>1</sub>* and *C<sub>2</sub>* in WordNet; the value of a dimension *H* in a vector is the information content (IC) of the WordNet hypernym *H*:

$$(1) \text{ IC}(H) = -\log\left(\frac{\text{size}(H)}{\sum_{c \in \text{WN}} \text{size}(c)}\right)$$

Here  $\text{size}(H)$  is the total number of lexical concepts in category  $H$  in WordNet, excluding any instance-level concepts, as these illustrative individuals are not evenly distributed across WordNet categories.

We also want any fine-grained perspective  $M-H$  to influence our similarity metric, provided it can be coherently tied into WordNet as a shared hypernym of the two lexical concepts being compared. The absolute information content of a category  $M-H$  that is newly added to WordNet is given by (2):

$$(2) \text{ IC}_{abs}(M-H) = -\log\left(\frac{\text{size}(M-H)}{\sum_{m-h \in \text{WN}} \text{size}(m-h)}\right)$$

where  $\text{size}(M-H)$  is the number of lexical concepts in WordNet for which  $M-H$  can be added as a new hypernym. The denominator in (2) denotes the sum total of the size of *all* fine-grained categories that can be coherently added to WordNet for *any* term.

The IC of  $M-H$  relative to  $H$  is estimated via the geometric mean of  $\text{IC}_{abs}(M-H)$  and  $\text{IC}(H)$  is given by (3):

$$(3) \text{ IC}(M-H) = \sqrt{\text{IC}_{abs}(M-H) \cdot \text{IC}(H)}$$

For a shared dimension  $H$  in the feature vectors of concepts  $C_1$  and  $C_2$ , if at least one fine-grained perspective  $M-H$  has been added to WordNet between  $H$  and  $C_1$  and between  $H$  and  $C_2$ , then the value of dimension  $H$  for  $C_1$  and for  $C_2$  is given by (4):

$$(4) \text{ weight}(H) = \max(\text{IC}(H), \max_M \text{IC}(M-H))$$

When no shared perspective  $M-H$  can be added under  $H$ , then  $\text{weight}(H) = \text{IC}(H)$ . A fine-grained perspective  $M-H$  will thus influence a similarity judgment between  $C_1$  and  $C_2$  only if  $M-H$  can be coherently added to WordNet as a hypernym of  $C_1$  and  $C_2$ , and if  $M-H$  enriches our view of  $H$ . Unlike Resnick (1995), Lin (1998) and Seco *et al.* (2006), this vector-space approach does not hinge on the information content of a single LCS, so any shared hypernym  $H$  or perspective  $M-H$  can shape a similarity judgment according to its informativeness.

## 5 Empirical Evaluation

Many fascinating perspectives on familiar ideas are bootstrapped from the web using similes as a starting point. These perspectives drive an exploratory web-aid to lateral thinking we call *Thesaurus Rex*, while the cosine-distance metric constructed from WordNet and these many fine-grained categories is called, simply, *Rex*. When *Rex* provides a numeric estimate of similarity for two ideas, *Thesaurus Rex* provides an enhanced insight into why these ideas are similar, e.g. by explaining that *cola & acid* are not just substances, they are *corrosive* substances.

We evaluate *Rex* by estimating how closely its judgments correlate with those of human judges on the 30-pair word set of Miller & Charles (M&C), who aggregated the judgments of multiple human raters into mean ratings for these pairs. We evaluate three variants of *Rex* on M&C: **Rex-lat**, which combines WordNet with all of *Thesaurus Rex*; **Rex-wn**, which uses only WordNet, with nothing at all from *Thesaurus Rex*; and **Rex-pop**, which enriches WordNet with only *popular* perspectives from *Thesaurus Rex*. A perspective is considered popular if it is discovered 5 or more times in the bootstrapping process, using 5 different anchors. While *corrosive-substance* is a popular category for *acid*, it not so for *cola* or *juice*. Popularity thus approximates what Ortony (1979) calls *salience*.

<i>Similarity metric</i>	<i>r</i>	<i>Similarity metric</i>	<i>r</i>
Wu & Palmer'94*	.74	Seco <i>et al.</i> '06*	.84
Resnick '95*	.77	Agirre <i>et al.</i> '09	<b>.93</b>
Leacock/Chod'98*	.82	Han <i>et al.</i> '09	.856
Lin '98*	.80	<b>Rex-wn</b>	.84
Jiang/Conrath '97*	-.81	<b>Rex-lat</b>	.89
Li <i>et al.</i> '03	.89	<b>Rex-pop</b>	<b>.93</b>

**Table 1.** Product-moment correlations (Pearson's  $r$ ) with mean human ratings on all 30 word pairs of the Miller & Charles similarity data-set.

\* As re-evaluated by Seco *et al.* (2006) for all 30 pairs

Table 1 lists coefficients of correlation (Pearson's  $r$ ) with mean human ratings for a range of WordNet-based metrics. Table 1 includes the hybrid *WordNet+web+SVM* metric of Agirre *et al.* (2009) – who report a correlation of **.93** – and the Mutual-Information-based *PMI<sub>max</sub>* metric of Han *et al.* (2009). The latter achieves good results for 27 of the 30 M&C pairs by enriching a PMI metric with an automatically-generated thesaurus. Yet while informative, this thesaurus is

not organized as an explanatory system of hierarchical categories as it is in *Thesaurus Rex*.

**Rex-wn** does no better than Seco *et al.* (2006) on the M&C dataset, suggesting that *Rex*'s vectors of IC-weighted hypernyms are no more discerning than a single informative LCS. However, such vectors also permit *Rex* to incorporate additional, fine-grained perspectives from *Thesaurus Rex*, allowing **Rex-lat** in turn to achieve a comparable correlation to that of Li *et al.* (2003) – .89. Yet the formulation in (2) favors unusual or idiosyncratic perspectives that are unlikely to generalize across independent judges. The mean ratings of M&C are the stuff of consensus, not individual creativity, and outside the realm of creative metaphor it often makes sense to safely align our judgments with those of others.

By limiting its use of *Thesaurus Rex* to the perspectives that other judges are most likely to use, **Rex-pop** obtains a correlation of .93 with mean human ratings on all 30 M&C pairs. This result is comparable to that reported by Agirre *et al.* (2009), who use SVM-based supervised learning to combine the judgments of two metrics, one based on WordNet and another on the analysis of web contexts of both input terms. However, *Rex* has the greater capacity for insight, since it augments the structured category system of WordNet with structured categories of its own. At each level of the WordNet hierarchy, *Rex* finds the fine-grained category that can best inform its judgments. Because *Rex* makes highly selective use of the diverse products of lateral thinking, this selectivity also produces concise explanations for its judgments.

## 5.1 Generative Uses of Similarity

A similarity metric offers a numerical measure of how closely one idea can cluster with another. It can also indicate how well one object may serve as a substitute for another, as when a *letter opener* is used as a *knife*, or *tofu* is used instead of *meat*. This need for substitution can be grist for creativity, yet most similarity metrics can only assess a suggested substitution, rather than suggest one for themselves. If they are to actively shape a creative decision, our similarity metrics must be made more generative.

A similarity metric can learn to be generative, by observing how people typically cluster words and ideas that are made similar by their contexts of use. The Google 3-grams contain many instances of the clustering pattern “*X+s and Y+s*”, as in “cowboys and pirates” or “doctors and lawyers”, and so a comprehensive trawl yields many

insights into the pairings of ideas that we implicitly see as comparable. We harvest all such Google 3-grams, to build a symmetric *comparability graph* in which any two comparable terms are adjacent nodes. For any node, we can generate a diverse set of comparable ideas just by reading off its adjacent nodes. *Thesaurus Rex* can be used to find an embracing category for many such pairs of nodes, while *Rex* estimates the similarity of any two adjacent nodes. A comparability graph of 28,000 nodes is produced from the Google 3-grams, with a sparse adjacency matrix of just 1,264,827 (0.16%) non-zero entries.

Is this dense enough for a task requiring generative similarity? Almuhareb & Poesio (2004) describe one such task: they sample 214 words from across 13 WordNet categories, and ask if these 214 words can be partitioned into 13 clusters that mirror the WordNet categories from which they were drawn. They then collect tens of thousands of web contexts for these 214 words, to extract a feature representation of each. We instead use *Rex* to generate, as features, a diverse set of comparable terms for each word. (We also assume that each word is a feature of itself). The *Rex* comparability graph suggests a pool of 8,300 features for all 214 words. The clustering toolkit CLUTO is used to partition the original 214 words into 13 clusters guided only by these comparability features. The resulting 13 clusters have an average purity of **93.4%** relative to WordNet, suggesting that categorization tasks which require implicit comparability judgments are well served by a generative approach to similarity.

## 5.2 Learning From Similarity Judgments

*Rex* augments the narrow worldview of WordNet with the more diverse viewpoints it gleans from the web, not by viewing them as separate knowledge sources, but by actually updating WordNet itself. The relative performance of **Rex-pop** > **Rex-lat** > **Rex-wn** on the M&C dataset shows that selective use of a divergent perspective permits WordNet to better serve its popular role as a judge of similarity. It is worth asking then whether these passing additions to WordNet should not be made permanent.

*Rex* estimates a similarity score for each of the 1,264,827 pairings of comparable terms it finds in the Google 3-grams. These scores are then cached to support generative similarity, and to permit fast lookup of scores for common comparisons. This lookup table is a lightweight means of using *Rex* in a range of creative substitution or generation tasks. Though the table is

sparse, §5.1 shows that it implicitly captures key nuances of category structure. The 39,826 unique fine-grained categories added by **Rex-pop** (versus the 44,238 categories added by **Rex-lat**) in the course of its 1,264,827 comparisons thus suggest credible enhancements to WordNet. Figure 2 graphs the distribution of new categories and their membership sizes when **Rex-pop** is used on this scale.

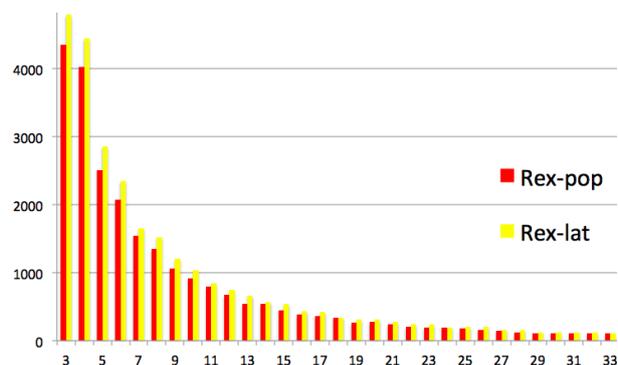


Figure 2. The number of new categories (Y-axis) with a given membership size (X-axis) added to WordNet when **Rex-pop/lat** are used on a large, web scale.

The *Goldilocks* categories are those that are not so small as to lack generality, and not so large as to lack information content. For example, **Rex-pop** suggests the addition of 15,125 new fine-grained categories to WordNet with membership sizes ranging from 5 to 25. This is a large but manageable number of categories that should be further considered for future addition to WordNet, or indeed to any similarly curated knowledge resource.

## 6 Summary and Conclusions

de Bono (1970) argues that the best solutions arise from using lateral and vertical thinking *in unison*. Lateral thinking is divergent and generative, while vertical thinking is convergent and analytical. The former can thus be used to create a pool of interesting candidates for the latter to selectively consider. *Thesaurus Rex* uses the web to generate a rich pool of alternate perspectives on familiar ideas, and *Rex* selects from this pool to perform vertical reasoning with WordNet to yield precise similarity judgments. *Rex* also uses the most informative perspective to concisely *explain* each comparison, or – when used in generative mode – to *suggest* a creative comparison. For instance, to highlight the potential toxicity of *coffee*, *Thesaurus Rex* suggests comparisons with *alcohol*, *tobacco* or *pesticide*, as all have been categorized as toxic substances on the web. A web app based on *Thesaurus Rex*, to support this

kind of lateral thinking, is accessible online at this URL:

<http://boundinanutshell.com/therex2>

Screenshots from the *Thesaurus Rex* application are provided in Figures 3 and 4 overleaf. Because *Thesaurus Rex* targets the acquisition of fine-grained perspectives, ranging from the off-beat to the obvious, it acquires an order-of-magnitude more categories from the web than can be found in WordNet itself. *Rex* dips selectively into this wealth of perspectives (and **Rex-pop** is more selective still), though many of *Rex*'s needs can be anticipated by looking to how ideas are implicitly grouped into *ad-hoc categories* (Barsalou, 1983) in constructions such as “*X+s and Y+s*”. Using the Google n-grams as a source of tacit grouping constructions, we have created a comprehensive lookup table that provides *Rex* similarity scores for the most common (if often implicit) comparisons.

Comparability is not the same as similarity, and a non-zero similarity score does not mean that two concepts would ever be considered comparable by a human. This poses a problem for the generation of sensible comparisons. However, *Rex*'s lookup table captures the implicit pragmatics of comparability, making *Rex* usable in generative tasks where a metric must both suggest and evaluate comparisons. Human similarity mechanisms are evaluative *and* generative, convergent *and* divergent. Our computational mechanisms should be no less so.

## 7 Acknowledgements

This research was partly supported by the WCU (World Class University) program under the National Research Foundation of Korea (Ministry of Education, Science and Technology of Korea, Project no. R31-30007) and partly funded by Science Foundation Ireland via the *Centre for Next Generation Localization* (CNGL).

subjective:skill, **personal:skill**, specialskill, subjective:measure, **personal:attribute**,  
basic:skill, **essential:skill**, **soft:skill**, professional:attribute, **mental:ability**, humanistic:attribute, **spiritual:attribute**,  
natural:attribute, **entrepreneurial:skill**, academic:ability, **subjective:thing**, musicatability, **important:attribute**, **key:skill**,  
**psychological:attribute**, **individual:skill**, natural:ability, **personal:motive**, **cognitive:skill**, **nonverbal:skill**,  
**important:skill**, **social:attribute**, **psychological:attitude**, **desirable:attribute**, **entrepreneurial:attribute**, **abstract:skill**,  
**intellectual:ability**, **social:skill**,  
**diverse:attribute**, cognitive:power, commercial:skill, technical:skill, mental:faculty, intellectual:skill, noble:attribute, artistic:skill, athletic:skill, feminine:attribute, feminine:skill,  
**wonderful:thing**, behavioural:skill, **individual:motivation**, **mental:attribute**, **positive:attribute**, vital:skill, interpersonal:skill,  
**professional:skill**, valuable:skill, **positive:attitude**, individual:attribute, **spiritual:power**, essential:attribute, **cognitive:ability**,  
**mental:skill**, educational:skill,

Figure 3. A screenshot from the web application *Thesaurus Rex*, showing the fine-grained categories found by *Thesaurus Rex* for the lexical concept *creativity* on the web.

**adverse\_event**, **bad\_event**, **bad\_thing**, **catastrophic\_event**, charged\_event, **changing\_event**,  
**critical\_event**, destructive\_thing,  
**devastating\_event**, **disruptive\_event**, **distressing\_event**, domestic\_conflict, domestic\_event,  
**dramatic\_event**,  
**economic\_event**, **emotional\_event**, environmental\_event, experienced\_event, **external\_event**,  
**extraordinary\_event**, financial\_event, identifiable\_event, immoral\_act, important\_event, intense\_event, **legal\_event**,  
major\_conflict, **major\_event**, **negative\_event**, ordinary\_event, outside\_event, painful\_event, **past\_event**,  
**rare\_event**, recent\_event, severe\_conflict,  
**severe\_event**,  
**significant\_event**, single\_event, social\_event, social\_occurrence, specified\_event, **stressful\_event**,  
**sudden\_event**, surrounding\_event, **traumatic\_event**, unanticipated\_event, unavoidable\_event,  
**uncontrollable\_event**, undesirable\_event,  
**unexpected\_event**, unexpected\_occurrence, **unforeseeable\_event**,  
**unforeseen\_event**, **unfortunate\_event**,  
**unpleasant\_event**,  
**unpleasant\_thing**, untoward\_event, **unusual\_event**,

Figure 4. A screenshot from the web application *Thesaurus Rex*, showing the shared overlapping categories found by *Thesaurus Rex* for the lexical concepts *divorce* and *war*.

## References

- Aristotle (translator: James Hutton). 1982. *Aristotle's Poetics*. New York: Norton.
- Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca and Aitor Soroa. 2009. Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches. In *Proceedings of NAACL '09, The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 19–27.
- Abdulrahman Almuhareb and Massimo Poesio. 2004. Attribute-Based and Value-Based Clustering: An Evaluation. In *Proceedings of the Conference on Empirical Methods in NLP*, Barcelona. pp. 158-165.
- Lawrence W. Barsalou. 1983. Ad hoc categories. *Memory and Cognition*, 11:211–227.
- Thorsten Brants and Alex Franz. 2006. *Web IT 5-gram Ver. 1*. Philadelphia: Linguistic Data Consortium.
- Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13-47.
- Mary K. Camac, and Sam Glucksberg. 1984. Metaphors do not use associations between concepts, they are used to create them. *Journal of Psycholinguistic Research*, 13, 443-455.
- de Bono, Edward. 1970. *Lateral thinking: creativity step by step*. New York: Harper & Row.
- de Bono, Edward. 1971. *Lateral thinking for management: a handbook for creativity*. New York: McGraw Hill.
- Christiane Fellbaum (ed.). 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.
- J. Paul Guilford. 1967. *The Nature of Human Intelligence*. New York: McGraw Hill.
- Lushan Han, Tim Finin, Paul McNamee, Anupam Joshi and Yelena Yesha. 2012. Improving Word Similarity by Augmenting PMI with Estimates of Word Polysemy. *IEEE Transactions on Data and Knowledge Engineering* (13 Feb. 2012).
- Yanfen Hao and Tony Veale. 2010. An Ironic Fist in a Velvet Glove: Creative Mis-Representation in the Construction of Ironic Similes. *Minds and Machines* 20(4), pp. 635–650.
- Jay Y. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10<sup>th</sup> International Conference on Research in Computational Linguistics*, pp. 19-33.
- Zornitsa Kozareva, Eileen Riloff and Eduard Hovy. 2008. Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. In *Proc. of the 46<sup>th</sup> Annual Meeting of the ACL*, pp 1048-1056.
- Claudia Leacock and Martin Chodorow. 1998. Combining local context and WordNet similarity for word sense identification. In Fellbaum, C. (ed.), *WordNet: An Electronic Lexical Database*, 265–283.
- Yuhua Li, Zuhair A. Bandar and David McLean. 2003. An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 4, pp. 871-882.
- Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15<sup>th</sup> ICML, the International Conference on Machine Learning*, Morgan Kaufmann, San Francisco CA, pp. 296–304.
- Michael Lesk. 1986 Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of ACM SigDoc, ACM*, 24–26.
- George A. Miller and Walter. G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes* 6(1):1-28.
- Andrew Ortony. 1979. Beyond literal similarity. *Psychological Review*, 86, pp. 161-180.
- Ted Pederson, Siddarth Patwardhan and Jason Michelizzi. 2004. WordNet::Similarity: measuring the relatedness of concepts. In *Proceedings of HLT-NAACL'04 (Demonstration Papers) the 2004 annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 38-41.
- Philip Resnick. 1995. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceedings of IJCAI'95, the 14<sup>th</sup> International Joint Conference on Artificial Intelligence*.
- Nuno Seco, Tony Veale and Jer Hayes, 2004. An Intrinsic Information Content Metric for Semantic Similarity in WordNet. In *Proceedings of ECAI'04, the European Conference on Artificial Intelligence*.
- Michael Strube and Simone Paolo Ponzetto. 2006. WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *Proceedings of AAAI-06, the 2006 Conference of the Association for the Advancement of AI*, pp. 1419–1424.
- Peter Turney. 2005. Measuring semantic similarity by latent relational analysis. *Proceedings of the 19<sup>th</sup> International Joint Conference on Artificial Intelligence*, 1136-1141.

- Tony Veale and Mark T. Keane. 1994. Belief Modeling, Intentionality and Perlocution in Metaphor Comprehension. In Proceedings of the 16<sup>th</sup> Annual Meeting of the Cognitive Science Society, Atlanta, Georgia. Hillsdale, NJ: Lawrence Erlbaum.
- Tony Veale. 2003. The analogical thesaurus: An emerging application at the juncture of lexical metaphor and information retrieval. In *Proceedings of IAAI'03, the 15<sup>th</sup> International Conference on Innovative Applications of Artificial Intelligence*, Mexico.
- Tony Veale. 2004. WordNet sits the SAT: A knowledge-based approach to lexical analogy. *Proceedings of ECAI'04, the European Conference on Artificial Intelligence*, 606-612.
- Tony Veale and Yanfen Hao. 2007. Comprehending and Generating Apt Metaphors: A Web-driven, Case-based Approach to Figurative Language. In proceedings of AAAI 2007, the 22nd AAAI Conference on Artificial Intelligence. Vancouver, Canada.
- Tony Veale, Guofu Li and Yanfen Hao. 2009. Growing Finely-Discriminating Taxonomies from Seeds of Varying Quality and Size. In *Proc. of EACL'09, the 12<sup>th</sup> Conference of the European Chapter of the Association for Computational Linguistics* pp. 835-842.
- Tony Veale. 2011. Creative Language Retrieval: A Robust Hybrid of Information Retrieval and Linguistic Creativity. In Proceedings of ACL'2011, the 49<sup>th</sup> Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.
- Tony Veale. 2012. Exploding the Creativity Myth: The computational foundations of linguistic creativity. *London: Bloomsbury Academic*.
- Tony Veale. 2013. Humorous Similes. *Humor: The International Journal of Humor Research*, **21**(1):3-22.
- Zhibiao Wu and Martha Palmer. 1994. Verb semantics and lexical selection. In *Proceedings of ACL'94, 32<sup>nd</sup> annual meeting of the Association for Computational Linguistics, Las Cruces, New Mexico*, pp. 133-138.

# Discovering User Interactions in Ideological Discussions

Arjun Mukherjee Bing Liu

Department of Computer Science

University of Illinois at Chicago

arjun4787@gmail.com liub@cs.uic.edu

## Abstract

Online discussion forums are a popular platform for people to voice their opinions on any subject matter and to discuss or debate any issue of interest. In forums where users discuss social, political, or religious issues, there are often heated debates among users or participants. Existing research has studied mining of user stances or camps on certain issues, opposing perspectives, and contention points. In this paper, we focus on identifying the nature of interactions among user pairs. The central questions are: How does each pair of users interact with each other? Does the pair of users mostly agree or disagree? What is the lexicon that people often use to express agreement and disagreement? We present a topic model based approach to answer these questions. Since agreement and disagreement expressions are usually multi-word phrases, we propose to employ a ranking method to identify highly relevant phrases prior to topic modeling. After modeling, we use the modeling results to classify the nature of interaction of each user pair. Our evaluation results using real-life discussion/debate posts demonstrate the effectiveness of the proposed techniques.

## 1 Introduction

Online discussion/debate forums allow people with common interests to freely ask and answer questions, to express their views and opinions on any subject matter, and to discuss issues of common interest. A large part of such discussions is about social, political, and religious issues. On such issues, there are often heated discussions/debates, i.e., people agree or disagree and argue with one another. Such ideological discussions on a myriad of social and political issues have practical implications in the fields of communication and political science as they give social scientists an opportunity to study real-life discussions/debates of almost any issue and analyze participant behaviors in a large scale.

In this paper, we present such an application, which aims to perform fine-grained analysis of user-interactions in online discussions.

There have been some related works that focus on discovering the general topics and ideological perspectives in online discussions (Ahmed and Xing, 2010), placing users in support/oppose camps (Agarwal et al., 2003), and classifying user stances (Somasundaran and Wiebe, 2009). However, these works are at a rather coarser level and have not considered more fine-grained characteristics of debates/discussions where users interact with each other by quoting/replying each other to express agreement or disagreement and argue with one another. In this work, we want to mine the following information:

1. The nature of interaction of each pair of users or participants who have engaged in the discussion of certain issues, i.e., whether the two persons mostly agree or disagree with each other in their interactions.
2. What language expressions are often used to express agreement (e.g., “I agree” and “you’re right”) and disagreement (e.g., “I disagree” and “you speak nonsense”).

We note that although agreement and disagreement expressions are distinct from traditional sentiment expressions (words and phrases) such as *good*, *excellent*, *bad*, and *horrible*, agreement and disagreement clearly express a kind of sentiment as well. They are usually emitted during interactive exchanges of arguments in ideological discussions. This idea prompted us to introduce the concept of *AD-sentiment*. We define the polarity of agreement expressions as *positive* and the polarity of disagreement expressions as *negative*. We refer agreement and disagreement expressions as *AD-sentiment expressions*, or *AD-expressions* for short. AD-expressions are crucial for the analysis of interactive discussions and debates just as sentiment expressions are instrumental in sentiment analysis (Liu, 2012). We thus regard this work as an extension to traditional sentiment

analysis (Pang and Lee, 2008; Liu, 2012).

In our earlier work (Mukherjee and Liu, 2012a), we proposed three topic models to mine contention points, which also extract AD-expressions. In this paper, we further improve the work by coupling an information retrieval method to rank good candidate phrases with topic modeling in order to discover more accurate AD-expressions. Furthermore, we apply the resulting AD-expressions to the new task of classifying the arguing or interaction nature of each pair of users. Using discovered AD-expressions for classification has an important advantage over traditional classification because they are domain independent. We employ a semi-supervised generative model called JTE-P to jointly model AD-expressions, pair interactions, and discussion topics simultaneously in a single framework. With such complex interactions mined, we can produce many useful summaries of discussions. For example, we can discover the most contentious pairs for each topic and ideological camps of participants, i.e., people who often agree with each other are likely to belong to the same camp. The proposed framework also facilitates tracking users' ideology shifts and the resulting arguing nature.

The proposed methods have been evaluated both qualitatively and quantitatively using a large number of real-life discussion/debate posts from four domains. Experimental results show that the proposed model is highly effective in performing its tasks and outperforms several baselines.

## 2 Related Work

There are several research areas that are related to our work. We compare with them below.

**Sentiment analysis:** Sentiment analysis determines positive and negative opinions expressed on entities and aspects (Hu and Liu, 2004). Main tasks include aspect extraction (Hu and Liu, 2004; Popescu and Etzioni, 2005), polarity identification (Hassan and Radev, 2010; Choi and Cardie, 2010) and subjectivity analysis (Wiebe, 2000). As discussed earlier, agreement and disagreement are a special form of sentiments and are different from the sentiment studied in the mainstream research. Traditional sentiment is mainly expressed with sentiment terms (e.g., *great* and *bad*), while agreement and disagreement are inferred by AD-expressions (e.g., *I agree* and *I disagree*), which we also call AD-sentiment expressions. Thus, this work expands the sentiment analysis research.

**Topic models:** Our work is also related to topic modeling and joint modeling of topics and other information as we jointly model several aspects of discussions/debates.

Topic models like pLSA (Hofmann, 1999) and LDA (Blei et al., 2003) have proved to be very successful in mining topics from large text collections. There have been various extensions to multi-grain (Titov and McDonald, 2008), labeled (Ramage et al., 2009), and sequential (Du et al., 2010) topic models. Yet other approaches extend topic models to produce author specific topics (Rosen-Zvi et al., 2004), author persona (Mimno and McCallum, 2007), social roles (McCallum et al., 2007), etc. However, these models do not model debates and hence are unable to discover AD-expressions and interaction natures of author pairs.

Also related are topic models in sentiment analysis which are often referred to as Aspect and Sentiment models (ASMs). ASMs come in two main flavors: Type-1 ASMs discover aspect (or topic) words sentiment-wise (i.e., discovering positive and negative topic words and sentiments for each topic without separating topic and sentiment terms) (e.g., Lin and He, 2009; Brody and Elhadad, 2010; Jo and Oh, 2011). Type-2 ASMs separately discover both aspects and sentiments (e.g., Mei et al., 2007; Zhao et al., 2010). Recently, domain knowledge induced ASMs have also been proposed (Mukherjee and Liu, 2012b; Chen et al., 2013). The generative process of ASMs is, however, different from our model. Specifically, Type-1 ASMs use asymmetric hyper-parameters for aspects while Type-2 assumes that sentiments and aspects are emitted in the same sentence. However, AD-expressions are emitted differently. They are mostly interleaved with users' topical viewpoints and span different sentences. Further, we capture the key characteristic of discussions by encoding pair-wise user interactions. Existing models do not model pair interactions.

In terms of discussions and comments, Yano et al., (2009) proposed the CommentLDA model which builds on the work of LinkLDA (Erosheva et al., 2004). Mukherjee and Liu (2012d) mined comment expressions. These works, however, don't model pair interactions in debates.

**Support/oppose camp classification:** Several works have attempted to put debate authors into support/oppose camps. Agrawal et al. (2003) used a graph based method. Murakami and Raymond (2010) used a rule-based method. In (Galley et al., 2004; Hillard et al., 2003), speaker

utterances were classified into agreement, disagreement and backchannel classes.

**Stances in online debates:** Somasundaran and Wiebe (2009), Thomas et al. (2006), Bansal et al. (2008), Burfoot et al. (2011), and Anand et al. (2011) proposed methods to recognize stances in online debates. Some other research directions include subgroup detection (Abu-Jbara et al., 2012), tolerance analysis (Mukherjee et al., 2013), mining opposing perspectives (Lin and Hauptmann, 2006), linguistic accommodation (Mukherjee and Liu, 2012c), and contention point mining (Mukherjee and Liu, 2012a). For this work, we adopt the JTE-P model in (Mukherjee and Liu, 2012a), and make two major advances. We propose a new method to improve the AD-expression mining and a new task of classifying pair interaction nature to determine whether each pair of users who have interacted based on replying relations mostly agree or disagree with each other.

### 3 Model

We now introduce the JTE-P model with additional details. JTE-P is a semi-supervised generative model motivated by the joint occurrence of expression types (*agreement* and *disagreement*), topics in discussion posts, and user pairwise interactions. Before proceeding, we make the following observation about online discussions.

In a typical debate/discussion post, the user (author) mentions a few topics (using semantically related topical terms) and expresses some viewpoints with one or more AD-expression types (using agreement and disagreement expressions). AD-expressions are directed towards other user(s), which we call *target*(s). In this work, we focus on explicit mentions (i.e., using @name or quoting other authors' posts). In our crawled dataset, 77% of all posts exhibit explicit quoting/reply-to relations excluding the first posts of threads which start the discussions and usually have nobody to quote/reply-to. Such author-target exchanges usually go back and forth between pairs of users populating a thread of discussion. The discussion topics and AD-expressions emitted are thus caused by the author-pairs' topical interests and their nature of interaction (agreeing vs. disagreeing).

In our discussion data obtained from Volconvo.com, we found that a pair of users typically exhibited a dominant arguing nature

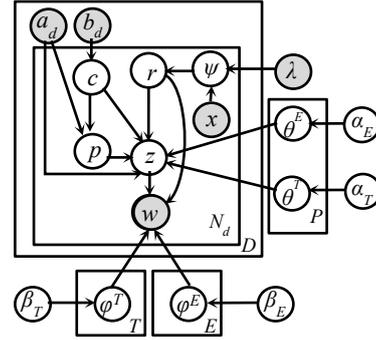


Figure 1: JTE-P Model in plate notation.

Variable/Function	Description
$d; a_d$	A document (post) $d$ ; author $a_d$ of document, $d$
$b_d = [b_1 \dots b_n]$	List of <i>targets</i> to whom $a_d$ replies/quotes in $d$ .
$p = (a, a')$	Pair of two authors interacting by reply/quote.
$\theta_p^T; \theta_p^E(\theta_{p,Ag}^E, \theta_{p,DisAg}^E)$	Pair $p$ 's distribution over topics; expression types (Agreement: $\theta_{p,Ag}^E$ , Disagreement: $\theta_{p,DisAg}^E$ )
$\varphi_t^T; \varphi_{e \in \{Ag, DisAg\}}^E$	Topic $t$ 's; Expression type $e$ 's distribution over vocabulary terms
$T; E$	Total number of topics; expression types
$V; P$	Total number of vocabulary terms; pairs
$w_{d,j}; N_d$	$j^{th}$ term in $d$ ; Total # of terms in $d$
$\psi_{d,j}$	Distribution over topics and AD-expressions
$x_{d,j}$	Associated feature context of the observed term $w_{d,j}$
$\lambda$	Learned Max-Ent parameters
$r_{d,j} \in \{\hat{t}, \hat{e}\}$	Binary indicator/switch variable (topic ( $\hat{t}$ ) or AD-expression ( $\hat{e}$ )) for $w_{d,j}$
$z_{d,j}$	Topic/Expression type of $w_{d,j}$
$\alpha_T; \alpha_E; \beta_T; \beta_E$	Dirichlet priors of $\theta_p^T; \theta_p^E; \varphi_t^T; \varphi_e^E$
$n_{p,t}^{PT}, n_{p,e}^{PE}$	# of times topic $t$ ; expression type $e$ assigned to $p$
$n_{t,v}^{CT}, n_{e,v}^{CE}$	# of times term $v$ appears in topic $t$ ; expression type $e$

Table 1: List of Notations

(agreeing vs. disagreeing) towards each other across various topics or threads. We believe this is because our data consists of topics like elections, theism, terrorism, vegetarianism, etc. which are often heated and attract people with pre-determined, strong, and polarized stances<sup>1</sup>.

This observation motivates the generative process of our model. Referring to the notations in Table 1, we explain the generative process of JTE-P. Given a document (post)  $d$ , its author,  $a_d$ , and the list of *targets* to whom  $a_d$  replies/quotes

<sup>1</sup> These hardened perspectives are supported by theoretical studies in communications like the polarization effect (Sunstein, 2002), and the hostile media effect, a scenario where partisans rigidly hold on to their stances (Hansen and Hyunjung, 2011).

in  $d$ ,  $b_d = [b_1 \dots b_n]$ , the document  $d$  exhibits shared topics and arguing nature of various pairs,  $p = (a_d, c)$ , where  $c \in b_d$ . More precisely, the pair specific topic and AD-expression distributions  $(\theta_p^T; \theta_p^E)$  “shape” the topics and AD-expressions emitted in  $d$  as agreement and disagreement on topical viewpoints are directed towards certain target authors. Each topic  $(\varphi_t^T)$  and AD-expression type  $(\varphi_e^E)$  is characterized by a multinomial distribution over terms (words/phrases). Assume we have  $t = 1 \dots T$  topics and  $e = 1 \dots E$  expression types in our corpus. Note that in our case of discussion/debate forums, we hypothesize  $E = 2$  as in debates, we mostly find two expression types: *agreement* and *disagreement* (more details in §6.1). Like most generative models for text, a post (document) is viewed as a bag of  $n$ -grams and each  $n$ -gram (word/phrase) takes one value from a predefined vocabulary. In this work, we use up to 4-grams, i.e.,  $n = 1, 2, 3, 4$ . Instead of using all  $n$ -grams, a relevance based ranking method is proposed to select a subset of highly relevant  $n$ -grams for model building (details in §4). For notational convenience, we use *terms* to denote both *words* (unigrams) and *phrases* ( $n$ -grams).

JTE-P is a switching graphical model (Ahmed and Xing, 2010; Zhao et al., 2010) performing a switch between AD-expressions and topics.  $\psi_{d,j}$  denotes the distribution over topics and AD-expressions with  $r_{d,j} \in \{\hat{t}, \hat{e}\}$  denoting the binary indicator/switch variable (topic or AD-expression) for the  $j^{\text{th}}$  term of  $d$ ,  $w_{d,j}$ . To perform the switch we use a maximum entropy (Max-Ent) model. The idea is motivated by the observation that topical and AD-expression terms usually play different roles in a sentence. Topical terms (e.g., “elections” and “income tax”) tend to be noun and noun phrases while AD-expression terms (“I refute”, “how can you say”, and “probably agree”) usually contain pronouns, verbs, wh-determiners, and modals. In order to utilize the part-of-speech (POS) tag information, we place the topic/AD-expression distribution  $\psi_{d,j}$  (the prior over the indicator variable  $r_{d,j}$ ) in the term plate (see Figure 1) and set it from a Max-Ent model conditioned on the observed feature context  $x_{d,j}$  associated with  $w_{d,j}$  and the learned Max-Ent parameters,  $\lambda$  (details in §6.1). In this work, we use both lexical and POS features of the previous, current, and next POS tags/lexemes of the term  $w_{d,j}$  as the contextual information, i.e.,  $x_{d,j} = [POS_{w_{d,j-1}}, POS_{w_{d,j}}, POS_{w_{d,j+1}}, w_{d,j-1}, w_{d,j}, w_{d,j+1}]$ , which is used to

produce the feature functions for Max-Ent. For phrasal terms ( $n$ -grams), all POS tags and lexemes of  $w_{d,j}$  are considered as contextual information for computing feature functions in Max-Ent. We now detail the generative process of JTE-P (plate notation in Figure 1) as follows:

1. For each AD-expression type  $e$ , draw  $\varphi_e^E \sim \text{Dir}(\beta_E)$
2. For each topic  $t$ , draw  $\varphi_t^T \sim \text{Dir}(\beta_T)$
3. For each pair  $p$ , draw  $\theta_p^E \sim \text{Dir}(\alpha_E)$ ;  $\theta_p^T \sim \text{Dir}(\alpha_T)$
4. For each forum discussion post  $d \in \{1 \dots D\}$ :
  - i. Given the author  $a_d$  and the list of targets  $b_d$ , for each term  $w_{d,j}$ ,  $j \in \{1 \dots N_d\}$ :
    - a. Draw a target  $c \sim \text{Uni}(b_d)$
    - b. Form pair  $p = (a_d, c)$ ,  $c \in b_d$
    - c. Set  $\psi_{d,j} \leftarrow \text{MaxEnt}(x_{d,j}; \lambda)$
    - d. Draw  $r_{d,j} \sim \text{Bern}(\psi_{d,j})$
    - e. if  $(r_{d,j} = \hat{e})$  //  $w_{d,j}$  is an AD-expression term  
Draw  $z_{d,j} \sim \text{Mult}(\theta_p^E)$   
else //  $r_{d,j} = \hat{t}$ ,  $w_{d,j}$  is a topical term  
Draw  $z_{d,j} \sim \text{Mult}(\theta_p^T)$
    - f. Emit  $w_{d,j} \sim \text{Mult}(\varphi_{z_{d,j}}^{r_{d,j}})$

*Dir*, *Mult*, *Bern*, and *Uni* correspond to the Dirichlet, Multinomial, Bernoulli, and Uniform distributions respectively. To learn JTE-P, we employ approximate posterior inference using Monte Carlo Gibbs sampling. Denoting the random variables  $\{w, z, p, r\}$  associated with each term by singular subscripts  $\{w_k, z_k, p_k, r_k\}$ ,  $k_{1..K}$ ,  $K = \sum_d N_d$ , a single Gibbs sweep consists of performing the following sampling.

$$p(z_k = t, p_k = p, r_k = \hat{t} | \dots) \propto \frac{1}{|b_d| \sum_{y \in \{\hat{e}, \hat{t}\}} \exp(\sum_{i=1}^n \lambda_i f_i(x_{d,j}, \hat{t}))} \times \frac{n_{p,t}^{PT} + \alpha_T}{n_{p,(\cdot)}^{PT} + T \alpha_T} \frac{n_{t,v}^{CT} + \beta_T}{n_{t,(\cdot)}^{CT} + V \beta_T} \quad (1)$$

$$p(z_k = e, p_k = p, r_k = \hat{e} | \dots) \propto \frac{1}{|b_d| \sum_{y \in \{\hat{e}, \hat{t}\}} \exp(\sum_{i=1}^n \lambda_i f_i(x_{d,j}, \hat{e}))} \times \frac{n_{p,e}^{PE} + \alpha_E}{n_{p,(\cdot)}^{PE} + E \alpha_E} \frac{n_{e,v}^{CE} + \beta_E}{n_{e,(\cdot)}^{CE} + V \beta_E} \quad (2)$$

Count variables  $n_{t,v}^{CT}$ ,  $n_{e,v}^{CE}$ ,  $n_{p,t}^{PT}$ , and  $n_{p,e}^{PE}$  are detailed in Table 1. Omission of a latter index denoted by  $(\cdot)$  represents the marginalized sum over the latter index.  $k = (d, j)$  denotes the  $j^{\text{th}}$  term of document  $d$  and the subscript  $-k$  denotes the counts excluding the term at  $(d, j)$ .  $\lambda_{1..n}$  are the parameters of the learned Max-Ent model corresponding to the  $n$  binary feature functions  $f_{1..n}$  for Max-Ent. These learned Max-Ent  $\lambda$  parameters in conjunction with the observed feature context,  $x_{d,j}$  feed the supervision signal for topic/expression switch parameter,  $r$  which is updated during inference in equations (1) and (2).

## 4 Phrase Ranking based on Relevance

We now detail our method of pre-processing  $n$ -grams (phrases) based on relevance to select a subset of highly relevant  $n$ -grams for model building. This has two advantages: (i). A large number of irrelevant  $n$ -grams slow inference. (ii). Filtering irrelevant terms in the vocabulary improves the quality of AD-expressions. Before proceeding, we review some existing approaches. Topics in most topic models like LDA are usually unigram distributions. This offers a great computational advantage compared to more complex models which consider word ordering (Wallach, 2006; Wang et al., 2007). This thread of research models bigrams by encoding them into the generative process. For each word, a topic is sampled first, then its status as a unigram or bigram is sampled, and finally the word is sampled from a topic-specific unigram or bigram distribution. This method, however, is expensive computationally and has a limitation for arbitrary length  $n$ -grams. In (Tomokiyo and Hurst, 2003), a language model approach is used for bigram phrase extraction.

Yet another thread of research *post-processes* the discovered topical unigrams to form multi-word phrases using likelihood scores (Blei and Lafferty, 2009). This approach considers adjacent word pairs and identifies  $n$ -grams which occur much more often than one would expect by chance alone by computing likelihood ratios. While this is reasonable, a significant  $n$ -gram with high likelihood score may *not* necessarily be relevant to the problem domain. For instance, in our case of discovering AD-expressions, the likelihood score<sup>2</sup> of  $p_1$  = “the government of” happens to be more than  $p_2$  = “I completely disagree”. Clearly, the former is irrelevant for the task of discovering AD-expressions. The reason for this is that likelihood scores or other statistical test scores rely on the relative counts in the multi-way contingency table to compute significance. Since the relative counts of different fragments of the irrelevant phrase  $p_1$ , e.g. “the government”, and “government of”, happen to appear more than the corresponding counts in the contingency table of  $p_2$ , the tests assign a higher score. This is nothing wrong per se because the statistical tests only judge significance of an  $n$ -gram, but a significant  $n$ -gram may not necessarily be relevant in a given problem domain.

<sup>2</sup> Computed using N-gram statistics package, NSP; <http://n-gram.sourceforge.net>

Thus, the existing approaches have some major shortcomings for our task. As our goal is to enhance the expressiveness of our models by considering relevant  $n$ -grams preserving the advantages of exchangeable modeling, we employ a *pre-processing* technique to rank  $n$ -grams based on relevance and consider certain number of top ranked  $n$ -grams based on *coverage* (details follow) in our vocabulary. The idea works as follows.

We first induce a unigram JTE-P whereby we cluster the relevant AD-expression unigrams in  $\phi_{Ag}^E$  and  $\phi_{DisAg}^E$ . Our notion of *relevance* of AD-expressions is already encoded into the model using priors set from Max-Ent. Next, we rank the candidate phrases ( $n$ -grams) using our probabilistic ranking function. The ranking function is grounded on the following hypothesis: a relevant phrase is one whose unigrams are closely related to (or appear with high probabilities in) the given AD-expression type,  $e$  : Agreement ( $Ag$ ) or disagreement ( $DisAg$ ). Continuing from the previous example, given the expression type  $\phi_{e=DisAg}^E$ ,  $p_2$  is relevant while  $p_1$  is not as “government” and “disagree” are highly unlikely and likely respectively to be clustered in  $\phi_{e=DisAg}^E$ . Thus, we want to rank phrases based on  $P(Rel = 1|e, p)$  where  $e$  denotes the expression type (Agreement/Disagreement),  $p$  denotes a candidate phrase. Following the probabilistic relevance model in (Lafferty and Zhai, 2003), we use a similar technique to that in (Zhao et al., 2011) for deriving our relevance ranking function as follows:

$$P(Rel = 1|e, p) = \frac{P(Rel=1|e, p)}{P(Rel=0|e, p) + P(Rel=1|e, p)} = \frac{1}{1 + \frac{P(Rel=0|e, p)}{P(Rel=1|e, p)}} = \frac{1}{1 + \frac{P(Rel=0, p|e)}{P(Rel=1, p|e)}} = \frac{1}{1 + \frac{[P(p|Rel=0, e) \times P(Rel=0|e)]}{[P(p|Rel=1, e) \times P(Rel=1|e)]}} \quad (3)$$

We further define  $\varepsilon = \frac{P(Rel=0|e)}{P(Rel=1|e)}$ . Without loss of generality, one can say that  $P(Rel = 0|e) \gg P(Rel = 1|e)$ , because there are many more irrelevant phrases than relevant ones, i.e.,  $\varepsilon \gg 1$ . Thus, taking log, from equation (3), we get,

$$\log P(Rel = 1|e, p) = \log \left( \frac{1}{1 + \varepsilon \times \frac{P(p|Rel=0, e)}{P(p|Rel=1, e)}} \right) \approx \log \left( \frac{P(p|Rel=1, e)}{P(p|Rel=0, e)} \times \frac{1}{\varepsilon} \right) = \log \left( \frac{P(p|Rel=1, e)}{P(p|Rel=0, e)} \right) - \log \varepsilon \quad (4)$$

Thus, our ranking function actually computes the relevance score  $\log \left( \frac{P(p|Rel=1, e)}{P(p|Rel=0, e)} \right)$ . The last term,  $\log \varepsilon$  being a constant is ignored because it cancels out while comparing candidate  $n$ -grams.

We now estimate the relevance score of a phrase  $p = (w_1, w_2, \dots, w_n)$ . Using the conditional independence assumption of words given the indicator variable  $Rel$  and expression type  $e$ , we have:

$$\log \left( \frac{P(p|Rel=1,e)}{P(p|Rel=0,e)} \right) = \sum_{i=1}^n \log \frac{P(w_i|Rel=1,e)}{P(w_i|Rel=0,e)} \quad (5)$$

Given the expression model  $\varphi_e^E$  previously learned by inducing the unigram JTE-P, it is intuitive to set  $P(w_i|Rel=1,e)$  to the point estimate of the posterior on  $\varphi_{e,w_i}^E = \frac{n_{e,w_i}^{EV} + \beta_E}{n_{e,(c)}^{EV} + V\beta_E}$ , where  $n_{e,w_i}^{EV}$  is the number of times  $w_i$  was assigned to AD-expression type  $e$  and  $n_{e,(c)}^{EV}$  denotes the marginalized sum over the latter index. On the other hand,  $P(w_i|Rel=0,e)$  can be estimated using a Laplace smoothed ( $\mu = 1$ ) background model, i.e.,  $(w_i|Rel=0,e) = \frac{n_{w_i} + \mu}{n_V + V\mu}$ , where  $n_{w_i}$  denotes the number of times  $w_i$  appears in the whole corpus and  $n_V$  denotes the number of terms in the entire corpus.

Next, we throw light on the issue of choosing the number of top  $k$  phrases from the ranked candidate  $n$ -grams. Precisely, we want to analyze the coverage of our proposed ranking based on relevance models. By coverage, we mean that having selected top  $k$  candidate  $n$ -grams based on the proposed relevance ranking, we want to get an estimate of how many relevant terms from a sample of the collection were covered. To compute coverage, we randomly sampled 500 documents from the corpus and listed the candidate  $n$ -grams<sup>3</sup> in the collection of sampled 500 documents. For this and subsequent human judgment tasks, we use two judges (graduate students well versed in English). We asked our judges to mark all relevant AD-expressions. Agreement study yielded  $\kappa_{\text{Cohen}} = 0.77$  showing substantial agreement according to scale<sup>4</sup> provided in (Landis and Koch, 1977). This is understandable as identifying AD-expressions is a relatively easy task. Finally, a term was considered to be relevant if both judges marked it so. We then computed the coverage to see how many of the relevant terms in the random sample were also present in top  $k$  phrases from the ranked candidate  $n$ -grams. We summarize the

<sup>3</sup> These are terms appearing at least 20 times in the entire collection. We do this for computational reasons as there can be many  $n$ -grams and  $n$ -grams with very low frequency are less likely to be relevant.

<sup>4</sup> No agreement ( $\kappa < 0$ ), slight agreement ( $0 < \kappa \leq 0.2$ ), fair agreement ( $0.2 < \kappa \leq 0.4$ ), moderate agreement ( $0.4 < \kappa \leq 0.6$ ), substantial agreement ( $0.6 < \kappa \leq 0.8$ ), and almost perfect agreement ( $0.8 < \kappa \leq 1.0$ ).

coverage results below in Table 2.

	$k$	3000	4000	5000
JTE-P	Agreement	81.34	84.24	87.01
	Disagreement	84.96	87.86	89.64

Table 2: Coverage (in %) of AD-expressions.

We find that choosing top  $k = 5000$  candidate  $n$ -grams based on our proposed ranking, we obtain a coverage of 87% for agreement and 89.64 for disagreement expression types which are reasonably good. Thus, we choose top 5000 candidate  $n$ -grams for each expression type and add them to the vocabulary beyond all unigrams.

Like expression types  $e_{1..E}$ , we also ranked candidate phrases for topics  $t_{1..T}$  using  $P(Rel=1|t,p)$ . However, for topics, selecting  $k$  based on coverage of each topic is more difficult because we induce 50 topics and it is also much more difficult to manually find relevant topical phrases in the sampled data as a topical phrase may belong to more than one topic. We selected top 2000 ranked candidate phrases for each topic using  $P(Rel=1|t,p)$  as we feel that is sufficient for a topic. Note that phrases for topics are not as crucial as for AD-expressions because topics can more or less be defined by unigrams.

## 5 Classifying Pair Interaction Nature

We now determine whether two users (also called a user pair) mostly agree or disagree with each other in their exchanges, i.e., their pair interaction or arguing nature. This is a relatively new task. We first summarize the closest related works. In (Galley et al., 2004; Hillard et al., 2003; Thomas et al., 2006, Bansal et al., 2008), conversational speeches (i.e., U.S. Congress meeting transcripts) are classified into for or against an issue using various types of features: *durational* (e.g., time taken by a speaker; speech rate, etc.), *structural* (e.g., no. of speakers per side, no. of votes cast by a speaker on a bill, etc.), and *lexical* (e.g., first word, last word,  $n$ -grams, etc.). Burfoot et al., (2011) builds on the work of (Thomas et al., 2006) and proposes collective classification using speaker contextual features (e.g., speaker intentions based on vote labels). However, above works do not discover pair interactions (arguing nature) in debate authors. Online discussion forums are textual rather than conversational (e.g., U.S. Congress meeting transcripts). Thus, the *durational*, *structural*, and *contextual* features used in prior works are not directly applicable.

Instead, the model posterior on  $\theta_p^E$  for JTE-P

can actually give an estimate of the overall interaction nature of a pair, i.e., the probability masses assigned to expression types,  $e = Ag$  (Agreement) and  $e = DisAg$  (Disagreement). As  $\theta_p^E \sim Dir(\alpha_E)$ , we have  $\theta_{p,e=Ag}^E + \theta_{p,e=DisAg}^E = 1$ . Hence, if the probability mass assigned to any one of the expression types (agreement, disagreement)  $> 0.5$  then according to the model posterior, that expression type is dominant, i.e., if  $\theta_{p,Ag}^E > 0.5$ , the pair is agreeing else disagreeing.

However, this approach is not the best. As we will see in the experiment section, supervised classification using labeled training data with discovered AD-expressions as features performs better.

## 6 Empirical Evaluation

We now evaluate the proposed techniques in the context of the JTE-P model. We first evaluate the discovered AD-expressions by comparing results with and without using the phrase ranking method in Section 4, and then evaluate the classification of interaction nature of pairs.

### 6.1 Dataset and Experiment Settings

We crawled debate/discussion forum posts from Volconvo.com. The forum is divided into various domains. Each domain consists of multiple threads of discussions. For each post, we extracted the post id, author, domain, ids of all posts to which it replies/quotes, and the post content. In all, we extracted 26137, 34986, 22354, and 16525 posts from Politics, Religion, Society and Science domains respectively.

**Experiment Data:** As it is not interesting to study pairs who only exchanged a few posts, we restrict to pairs with at least 20 post exchanges. This resulted in 1241 authors and 1461 pairs. The reduced dataset consists of 1095586 tokens (after  $n$ -gram preprocessing in §4), 40102 posts with an average of 27 posts or interactions per pair. Data from all 4 domains are combined for modeling.

**Parameter Settings:** For all our experiments, we set the hyper-parameters to the heuristic values  $\alpha_T = 50/T$ ,  $\alpha_E = 50/E$ ,  $\beta_T = \beta_E = 0.1$  suggested in (Griffiths and Steyvers, 2004). We set the number of topics,  $T = 50$  and the number of AD-expression types,  $E = 2$  (agreement and disagreement) as in discussion/debate forums, there are usually two expression types<sup>5</sup>. To learn

<sup>5</sup> Values for  $E > 2$  were also tried. However, they did not produce any new dominant expression type. There was also a slight increase in the model perplexity showing that values of  $E > 2$  do not fit the debate forum data well.

the Max-Ent parameters  $\lambda$ , we randomly sampled 500 terms from the held-out data (10 threads in our corpus which were excluded from the evaluation of tasks in §6.2, §6.3) appearing at least 10 times and labeled them as topical (361) or AD-expressions (139) and used the corresponding features of each term (in the context of posts where it occurs, §3) to train the Max-Ent model.

### 6.2 AD-Expression Evaluation

We first list some discovered top AD-expressions in Table 3 for qualitative inspection. From Table 3, we can see that JTE-P can cluster many correct AD-expressions, e.g., “I accept”, “I agree”, “you’re correct”, etc. in agreement and “I disagree”, “don’t accept”, “I refute”, etc. in disagreement. In addition, it also discovers and clusters highly specific and more “distinctive” expressions beyond those used in Max-Ent training, e.g., “valid point”, “I do support”, and “rightly said” in agreement; and phrases like “can you prove”, “I don’t buy your”, and “you fail to” in disagreement. Note that terms in black in Table 3 were used in Max-Ent training. The newly discovered terms are marked *blue* in italics. Clustering errors are in **red** (bold).

For quantitative evaluation, topic models are often compared using perplexity. However, perplexity does not reflect our purpose since we are not trying to evaluate how well the AD-expressions in an unseen discussion data fit our learned models. Instead our focus is to evaluate how well our learned AD-expression types perform in clustering semantic phrases of agreement/disagreement. Since AD-expressions (according to top terms in  $\phi^E$ ) produced by JTE-P are rankings, we choose *precision @ n* ( $p@n$ ) as our metric.  $p@n$  is commonly used to evaluate a ranking when the total number of correct items is unknown (e.g., Web search results, aspect terms in topic models for sentiment analysis (Zhao et al., 2010), etc.). This situation is similar to our AD-expression rankings,  $\phi^E$ . Further, as  $\phi^E \sim Dir$ , the Dirichlet smoothing effect ensures that every term in the vocabulary has some non-zero mass to agreement or disagreement expression type. Thus, it is the ranking of terms in each AD-expression type that matters (i.e., whether the model is able to rank highly relevant terms at the top).

The above method evaluates the original ranking. Another way of evaluating the AD-expression rankings is to evaluate only those newly discovered terms, i.e., beyond those

Disagreement expressions ( $\varphi_{e=Disagreement}^E$ )	
I, disagree, I don't, I disagree, <b>argument</b> , reject, <b>claim</b> , I reject, I refute, <b>and</b> , <b>your</b> , I refuse, <b>won't</b> , <b>the claim</b> , nonsense, <i>I contest</i> , dispute, <b>I think</b> , completely disagree, don't accept, don't agree, incorrect, <b>doesn't</b> , <i>hogwash</i> , <i>I don't buy your</i> , <i>I really doubt</i> , your nonsense, <b>true</b> , <i>can you prove</i> , argument fails, <i>you fail to</i> , <b>your assertions</b> , <i>bullshit</i> , <i>sheer nonsense</i> , <i>doesn't make sense</i> , <i>you have no clue</i> , <i>how can you say</i> , <i>do you even</i> , <i>contradict yourself</i> , ...	
Agreement expressions ( $\varphi_{e=Agreement}^E$ )	
agree, <b>I</b> , correct, yes, true, accept, I agree, <b>don't</b> , indeed correct, <b>your</b> , I accept, <b>point</b> , <b>that</b> , I concede, is valid, <b>your claim</b> , <b>not really</b> , <i>would agree</i> , <b>might</b> , <i>agree completely</i> , yes indeed, absolutely, you're correct, <i>valid point</i> , <b>argument</b> , <b>the argument</b> , proves, <i>do accept</i> , support, agree with you, <i>rightly said</i> , <b>personally</b> , well put, <i>I do support</i> , <i>personally agree</i> , <b>doesn't necessarily</b> , exactly, <i>very well put</i> , <i>kudos</i> , <i>point taken</i> , ...	

Table 3: Top terms (comma delimited) of two expression types. **Red** (bold) terms denote possible errors. *Blue* (italics) terms are newly discovered; rest (black) terms have been used in Max-Ent training.

$P@n$	JTE-P (all terms)						JTE-P (excluding labeled ME terms)					
	Agreement			Disagreement			Agreement			Disagreement		
	50	100	150	50	100	150	50	100	150	50	100	150
100	0.62	0.63	0.61	0.64	0.62	0.63	0.58	0.56	0.57	0.60	0.59	0.58
200	0.66	0.67	0.65	0.68	0.66	0.67	0.62	0.59	0.60	0.64	0.63	0.62
300	0.70	0.70	0.71	0.70	0.68	0.67	0.66	0.66	0.65	0.66	0.66	0.65
400	0.72	0.72	0.73	0.74	0.71	0.70	0.68	0.67	0.69	0.70	0.68	0.69
500	0.76	0.77	0.75	0.76	0.73	0.74	0.70	0.71	0.70	0.72	0.71	0.70

Table 4: Results using terms based on phrase relevance ranking for  $P@n=50, 100, 150$  across 100, 200, ..., 500 labeled examples ( $L$ ) used for Max-Ent (ME) training.

$P@n$	JTE-P (all terms)						JTE-P (excluding ME terms)					
	Agreement			Disagreement			Agreement			Disagreement		
	50	100	150	50	100	150	50	100	150	50	100	150
500	0.66	0.69	0.69	0.72	0.70	0.70	0.66	0.65	0.64	0.68	0.66	0.65

Table 5: Results using all tokens (without applying phrase relevance ranking) for  $P@50, 100, 150$  and 500 labeled examples were used for Max-Ent (ME) training.

Feature Setting	Agreeing			Disagreeing		
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>
JTE-P-posterior	0.59	0.61	0.60	0.81	0.70	0.75
W+POS 1-4 grams	0.63	0.66	0.64	0.83	0.82	0.82
W+POS 1-4grams + IG (top 1%)	0.64	0.67	0.65	0.84	0.82	0.83
W+POS 1-4 grams + IG (top 2%)	0.65	0.67	0.66	0.84	0.82	0.83
W+POS 1-4 grams + $\chi^2$ (top 1%)	0.65	0.68	0.66	0.84	0.83	0.83
W+POS 1-4 grams + $\chi^2$ (top 2%)	0.64	0.68	0.69	0.84	0.82	0.83
AD-Expressions, $\Phi^E$ (top 1000)	0.73	0.74	0.73	0.87	0.87	0.87
AD-Expressions, $\Phi^E$ (top 2000)	0.77	0.81	0.78	0.90	0.88	0.89

Table 6: Precision (P), recall (R), and F<sub>1</sub> scores of pair interaction evaluation. Improvements in F<sub>1</sub> using AD-expression features ( $\varphi^E$ ) are statistically significant ( $p<0.01$ ) using paired  $t$ -test across 5-fold CV.

labeled terms used in Max-Ent training. For this evaluation, we remove those terms that have been used in Max-Ent (ME) training. We report both results in Table 4. We also studied inter-rater agreement using two judges who independently labeled the top  $n$  terms as correct or incorrect. A term was marked correct if both judges deemed it so which was then used to compute  $p@n$ . Agreement using  $\kappa_{Cohen}$  was greater than 0.78 for all  $p@n$  computations implying substantial and good agreements as identifying whether a phrase implies agreement or disagreement or none is an easy task.  $P@n$  excluding ME labeled terms (Table 4, second

column) are slightly lower than those using all terms but are still decent. This is because  $p@n$  excluding ME labeled terms removes many correct AD-expressions used in training.

Further to evaluate the sensitivity of performance on the amount of labeled terms for Max-Ent, we computed  $p@n$  across different sizes of labeled terms. Table 4 shows  $p@n$  for agreement and disagreement expressions across different sizes of labeled terms ( $L$ ). We find that more labeled terms improves  $p@n$  which is intuitive. We used 500 labeled terms in all our subsequent experiments. The result in Table 4 uses relevance ranking (§4).

We now compare with the performance of the model without using phrase relevance ranking.  $P@n$  results using all tokens (4356787) are shown in Table 5 (with 500 labeled terms for Max-Ent training). Clearly,  $P@n$  is lower than in Table 4 (last row; with phrase relevance ranking) because without phrase relevance ranking (Table 5) many irrelevant terms can rank high due to co-occurrences which may not be semantically related. This shows that relevance ranking of phrases is beneficial.

### 6.3 Pair Interaction Nature

We now evaluate the overall interaction nature of each pair of users. The evaluation of this task requires human judges to read all the posts where the two users forming the pair have interacted. Thus, it is hard to evaluate all 1461 pairs in our dataset. Instead, we randomly sampled 500 pairs ( $\approx 34\%$  of the population) for evaluation. Two human judges were asked to independently read all the post interactions of 500 pairs and label each pair as overall “disagreeing” or overall “agreeing” or “none”. The  $\kappa_{Cohen}$  for this task was 0.81. Pairs were finally labeled as agreeing or disagreeing if both judges deemed them so. This resulted in 320 disagreeing and 152 agreeing pairs. Out of the rest 28 pairs, 10 were marked “none” by both judges while 18 pairs had disagreement in labels. We only focus on the 472 agreeing and disagreeing pairs.

As we have labeled data for 472 pairs, we can treat identifying pair arguing nature as a text classification problem where all interactions between a pair are merged in one document representing the pair along with the label given by judges: agreeing or disagreeing. To compare classification performance, we use two feature sets: (i) standard word + POS 1-4 grams and (ii) AD-expressions from  $\varphi^E$ . We use TF-IDF as our feature value assignment scheme. We also try two well-known feature selection schemes Chi-Squared Test ( $\chi^2$ ) and Information Gain (IG). We use the linear kernel<sup>6</sup> SVM (SVM<sup>light</sup> system in (Joachims, 1999)) as our text classifier. For feature selection using  $\chi^2$  and IG, we use two settings: top 1% and 2% of all features ranked according to the selection metric. Also, for estimated AD-expressions (according to probabilities in  $\varphi^E$ ), we experiment with top 1000 and 2000 AD-expressions terms for both agreement and disagreement. We summarize

<sup>6</sup> Other kernels polynomial, RBF, and sigmoid did not perform as well.

comparison results using 5-fold Cross Validation (CV) with two classes: agreeing and disagreeing in Table 6. JTE-P-posterior represents the method using simply the model posterior on  $\theta_p^E$  to make the decision (see §5). From Table 6, we can make the following observations.

Predicting agreeing arguing nature is harder than that of disagreeing across all feature settings. Feature selection improves performance.  $\chi^2$  and IG perform similarly. AD-expressions,  $\varphi^E$  yields the best performance showing that the discovered AD-expressions are of high quality and reflect the user pair arguing nature well. Selecting certain top terms in  $\varphi^E$  can also be viewed as a form of feature selection. Although prediction performance using model posterior (JTE-P-posterior) is slightly lower than supervised SVM (Table 6, second row), the  $F_1$  scores are decent. Using the discovered AD-expressions (Table 6, last row) as features renders a statistically significant (see Table 6 caption) improvement over other baseline feature settings. This shows that discovered AD-expressions are useful for downstream applications, e.g., the task of identifying pair interactions.

## 7 Conclusion

This paper studied the problem of modeling user pair interactions in online discussions with the purpose of discovering the interaction or arguing nature of each author pair and various AD-expressions emitted in debates. A novel technique was also proposed to rank  $n$ -gram phrases where relevance based ranking was used in conjunction with a semi-supervised generative model. This method enables us to find better AD-expressions. Experiments using real-life online debate data showed the effectiveness of the model. In our future work, we intend to extend the model to account for stances, and issue specific interactions which would pave the way for user profiling and behavioral modeling.

## Acknowledgments

We would like thank Sharon Meraz (Department of communication, University of Illinois at Chicago) and Dennis Chong (Department of Political Science, Northwestern University) for several valuable discussions. This work was supported in part by a grant from the National Science Foundation (NSF) under grant no. IIS-1111092.

## References

- Abu-Jbara, A., Dasigi, P., Diab, M. and Dragomir Radev. 2012. Subgroup detection in ideological discussions. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2012).
- Agrawal, R., Rajagopalan, S., Srikant, R., and Xu. Y. 2003. Mining newsgroups using networks arising from social behavior. In Proceedings of the International Conference on World Wide Web (WWW-2003).
- Ahmed, A and Xing, E. 2010. Staying informed: supervised and semi-supervised multi-view topical analysis of ideological perspective. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP-2010).
- Anand, P., Walker, M., Abbott, R., Tree, J., Bowmani, R., and Minor, M. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis.
- Bansal, M., Cardie, C., and Lee, L. 2008. The power of negative thinking: Exploiting label disagreement in the min-cut classification framework. In Proceedings of the International Conference on Computational Linguistics (Short Paper).
- Blei, D., Ng, A., and Jordan, M. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*.
- Blei, D. and Lafferty J. 2009. Visualizing topics with multi-word expressions. Tech. Report. arXiv:0907.1013v1.
- Brody, S. and Elhadad, S. 2010. An Unsupervised Aspect-Sentiment Model for Online Reviews. In Proceedings of the Annual Conference of the North American Chapter of the ACL (NAACL-2010).
- Burfoot, C., Bird, S., and Baldwin, T. 2011. Collective Classification of Congressional Floor-Debate Transcripts. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2011).
- Chang, J., Boyd-Graber, J., Wang, C. Gerrish, S. Blei, D. 2009. Reading tea leaves: How humans interpret topic models. In Proceedings of the Neural Information Processing Systems (NIPS-2009).
- Chen, Z., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M., Ghosh, R. 2013. Leveraging Multi-Domain Prior Knowledge in Topic Models. In Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI-2013).
- Choi, Y. and Cardie, C. 2010. Hierarchical sequential learning for extracting opinions and their attributes (Short Paper). In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2010).
- Du, L., Buntine, W. L., and Jin, H. 2010. Sequential Latent Dirichlet Allocation: Discover Underlying Topic Structures within a Document. In Proceedings of the IEEE International Conference on Data Mining (ICDM-2010).
- Erosheva, E., Fienberg, S. and Lafferty, J. 2004. Mixed membership models of scientific publications. In Proceedings of the National Academy of Sciences (PNAS-2004).
- Galley, M., McKeown, K., Hirschberg, J., and Shriberg, E. 2004. Identifying agreement and disagreement in conversational speech: Use of Bayesian networks to model pragmatic dependencies. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2004).
- Griffiths, T. and Steyvers, M. 2004. Finding scientific topics. In Proceedings of the National Academy of Sciences (PNAS-2004).
- Hansen, G. J., and Hyunjung, K. 2011. Is the media biased against me? A meta-analysis of the hostile media effect research. *Communication Research Reports*, 28, 169-179.
- Hillard, D., Ostendorf, M., and Shriberg, E. 2003. Detection of agreement vs. disagreement in meetings: Training with unlabeled data. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2003).
- Hassan, A. and Radev, D. 2010. Identifying text polarity using random walks. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2010).
- Hofmann, T. 1999. Probabilistic latent semantic analysis. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI-1999).
- Hu, M. and Liu, B. 2004. Mining and summarizing customer reviews. In Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004).
- Jo, Y. and Oh, A. 2011. Aspect and sentiment unification model for online review analysis. In Proceedings of the International Conference on Web Search and Data Mining (WSDM-2011).
- Joachims, T. Making large-Scale SVM Learning Practical. 1999. *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.
- Lafferty, J. and Zhai, C. 2003. Probabilistic relevance models based on document and query generation. *Language Modeling and Information Retrieval*.
- Landis, J. R. and Koch, G. G. 1977. The measurement of observer agreement for categorical data. *Biometrics*.
- Lin, C. and He, Y. 2009. Joint sentiment/topic model for sentiment analysis. In Proceedings of the

- International Conference on Knowledge Management (CIKM-2009).
- Lin, W. H., and Hauptmann, A. 2006. Are these documents written from different perspectives?: a test of different perspectives based on statistical distribution divergence. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2006).
- Liu, B. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publisher, USA.
- McCallum, A., Wang, X., and Corrada-Emmanuel, A. 2007. Topic and Role Discovery in Social Networks with Experiments on Enron and Academic Email. *Journal of Artificial Intelligence Research*.
- Mei, Q., Ling, X., Wondra, M., Su, H., and Zhai, C. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In Proceedings of the International Conference on World Wide Web (WWW-2007).
- Mimno, D. and McCallum, A. 2007. Expertise modeling for matching papers with reviewers. In Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2007).
- Mukherjee, A., Venkataraman, V., Liu, B., Meraz, S. 2013. Public Dialogue: Analysis of Tolerance in Online Discussions. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2013).
- Mukherjee, A. and Liu, B. 2012a. Mining Contentions from Discussions and Debates. Proceedings of SIGKDD Conference on Knowledge Discovery and Data Mining (KDD-2012).
- Mukherjee, A. and Liu, B. 2012b. Aspect Extraction through Semi-Supervised Modeling. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2012).
- Mukherjee, A. and Liu, B. 2012c. Analysis of Linguistic Style Accommodation in Online Debates. In Proceedings of the International Conference on Computational Linguistics (COLING-2012).
- Mukherjee, A. and Liu, B. 2012d. Modeling Review Comments. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2012).
- Murakami A. and Raymond, R. 2010. Support or Oppose? Classifying Positions in Online Debates from Reply Activities and Opinion Expressions. In Proceedings of the International Conference on Computational Linguistics (Coling-2010).
- Pang, B. and Lee, L. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*.
- Popescu, A. and Etzioni, O. 2005. Extracting product features and opinions from reviews. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2005).
- Ramage, D., Hall, D., Nallapati, R., Manning, C. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2009).
- Rosen-Zvi, M., Griffiths, T., Steyvers, M., and Smith, P. 2004. The author-topic model for authors and documents. In Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI-2004).
- Sunstein, C. R. 2002. The law of group polarization. *Journal of political philosophy*.
- Somasundaran, S. and Wiebe, J. 2009. Recognizing stances in online debates. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP-2009).
- Titov, I. and R. McDonald. 2008. Modeling online reviews with multi-grain topic models. In Proceedings of the International Conference on World Wide Web (WWW-2008).
- Thomas, M., Pang, B., and Lee, L. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2006).
- Tomokiyo, T., and Hurst, M. 2003. A language model approach to keyphrase extraction. In Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-Volume 18.
- Wallach, H. 2006. Topic modeling: Beyond bag of words. In Proceedings of the International Conference on Machine Learning (ICML-2006).
- Wang, X., McCallum, A., Wei, X. 2007. Topical N-grams: Phrase and topic discovery, with an application to information retrieval. In Proceedings of the IEEE International Conference on Data Mining (ICDM-2007).
- Wiebe, J. 2000. Learning subjective adjectives from corpora. In Proc. of National Conference on AI (AAAI-2000).
- Yano, T., Cohen, W. and Smith, N. 2009. Predicting response to political blog posts with topic models. In Proceedings of the N. American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2009).
- Zhao, X., J. Jiang, J. He, Y. Song, P. Achananuparp, E.P. LiM, and X. Li. 2011. Topical keyphrase extraction from twitter. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-2011).
- Zhao, X., Jiang, J., Yan, H., and Li, X. 2010. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2010).

# Multilingual Affect Polarity and Valence Prediction in Metaphor-Rich Texts

Zornitsa Kozareva

USC Information Sciences Institute  
4676 Admiralty Way  
Marina del Rey, CA 90292-6695  
kozareva@isi.edu

## Abstract

Metaphor is an important way of conveying the affect of people, hence understanding how people use metaphors to convey affect is important for the communication between individuals and increases cohesion if the perceived affect of the concrete example is the same for the two individuals. Therefore, building computational models that can automatically identify the affect in metaphor-rich texts like “*The team captain is a rock.*”, “*Time is money.*”, “*My lawyer is a shark.*” is an important challenging problem, which has been of great interest to the research community.

To solve this task, we have collected and manually annotated the affect of metaphor-rich texts for four languages. We present novel algorithms that integrate triggers for cognitive, affective, perceptual and social processes with stylistic and lexical information. By running evaluations on datasets in English, Spanish, Russian and Farsi, we show that the developed affect polarity and valence prediction technology of metaphor-rich texts is portable and works equally well for different languages.

## 1 Introduction

Metaphor is a figure of speech in which a word or phrase that ordinarily designates one thing is used to designate another, thus making an implicit comparison (Lakoff and Johnson, 1980; Martin, 1988; Wilks, 2007). For instance, in

“*My lawyer is a shark*”

the speaker may want to communicate that his/her lawyer is strong and aggressive, and that he will

attack in court and persist until the goals are achieved. By using the metaphor, the speaker actually conveys positive affect because having an aggressive lawyer is good if one is being sued.

There has been a substantial body of work on metaphor identification and interpretation (Wilks, 2007; Shutova et al., 2010). However, in this paper we focus on an equally interesting, challenging and important problem, which concerns the automatic identification of affect carried by metaphors. Building such computational models is important to understand how people use metaphors to convey affect and how affect is expressed using metaphors. The existence of such models can be also used to improve the communication between individuals and to make sure that the speakers perceived the affect of the concrete metaphor example in the same way.

The questions we address in this paper are: “*How can we build computational models that can identify the polarity and valence associated with metaphor-rich texts?*” and “*Is it possible to build such automatic models for multiple languages?*”. Our main contributions are:

- We have developed multilingual metaphor-rich datasets in English, Spanish, Russian and Farsi that contain annotations of the *Positive* and *Negative* polarity and the valence (from  $-3$  to  $+3$  scale) corresponding to the intensity of the affect conveyed in the metaphor.
- We have proposed and developed automated methods for solving the polarity and valence tasks for all four languages. We model the polarity task as a classification problem, while the valence task as a regression problem.
- We have studied the influence of different information sources like the metaphor itself, the context in which it resides, the source and

target domains of the metaphor, in addition to contextual features and trigger word lists developed by psychologists (Tausczik and Pennebaker, 2010).

- We have conducted in depth experimental evaluation and showed that the developed methods significantly outperform baseline methods.

The rest of the paper is organized as follows. Section 2 describes related work, Section 3 briefly talks about metaphors. Sections 4 and 5 describe the polarity classification and valence prediction tasks for affect of metaphor-rich texts. Both sections have information on the collected data for English, Spanish, Russian and Farsi, the conducted experiments and obtained results. Finally, we conclude in Section 6.

## 2 Related Work

A substantial body of work has been done on determining the affect (sentiment analysis) of texts (Kim and Hovy, 2004; Strapparava and Mihalcea, 2007; Wiebe and Cardie, 2005; Yessenalina and Cardie, 2011; Breck et al., 2007). Various tasks have been solved among which polarity and valence identification are the most common. While polarity identification aims at finding the *Positive* and *Negative* affect, valence is more challenging as it has to map the affect on a  $[-3, +3]$  scale depending on its intensity (Polanyi and Zaenen, 2004; Strapparava and Mihalcea, 2007).

Over the years researchers have developed various approaches to identify polarity of words (Esuli and Sebastiani, 2006), phrases (Turney, 2002; Wilson et al., 2005), sentences (Choi and Cardie, 2009) even documents (Pang and Lee, 2008). Multiple techniques have been employed, from various machine learning classifiers, to clustering and topic models. Various domains and textual sources have been analyzed such as Twitter, Blogs, Web documents, movie and product reviews (Turney, 2002; Kennedy and Inkpen, 2005; Niu et al., 2005; Pang and Lee, 2008), but yet what is missing is affect analyzer for metaphor-rich texts.

While the affect of metaphors is well studied from its linguistic and psychological aspects (Blanchette et al., 2001; Tomlinson and Love, 2006; Crowdord, 2009), to our knowledge the building of computational models for polarity and valence identification in metaphor-rich texts is still

a novel task (Smith et al., 2007; Veale, 2012; Veale and Li, 2012; Reyes and Rosso, 2012; Reyes et al., 2013). Little (almost no) effort has been put into multilingual computational affect models of metaphor-rich texts. Our research specifically targets the resolution of these problems and shows that it is possible to build such computational models. The experimental result provide valuable contributions and fundings, which could be used by the research community to build upon.

## 3 Metaphors

Although there are different views on metaphor in linguistics and philosophy (Black, 1962; Lakoff and Johnson, 1980; Gentner, 1983; Wilks, 2007), the common among all approaches is the idea of an interconceptual mapping that underlies the production of metaphorical expressions. There are two concepts or conceptual domains: the target (also called topic in the linguistics literature) and the source (or vehicle), and the existence of a link between them gives rise to metaphors.

The texts “*Your claims are **indefensible**.*” and “*He **attacked** every weak point in my argument.*” do not directly talk about argument as a war, however the winning or losing of arguments, the attack or defense of positions are structured by the concept of war. There is no physical battle, but there is a verbal battle and the structure of an argument (attack, defense) reflects this (Lakoff and Johnson, 1980).

As we mentioned before, there has been a lot of work on the automatic identification of metaphors (Wilks, 2007; Shutova et al., 2010) and their mapping into conceptual space (Shutova, 2010a; Shutova, 2010b), however these are beyond the scope of this paper. Instead we focus on an equally interesting, challenging and important problem, which concerns the automatic identification of affect carried by metaphors. To conduct our study, we use human annotators to collect metaphor-rich texts (Shutova and Teufel, 2010) and tag each metaphor with its corresponding polarity (*Positive/Negative*) and valence  $[-3, +3]$  scores. The next sections describe the affect polarity and valence tasks we have defined, the collected and annotated metaphor-rich data for each one of the English, Spanish, Russian and Farsi languages, the conducted experiments and obtained results.

## 4 Task A: Polarity Classification

### 4.1 Problem Formulation

**Task Definition:** Given metaphor-rich texts annotated with *Positive* and *Negative* polarity labels, the goal is to build an automated computational affect model, which can assign to previously unseen metaphors one of the two polarity classes.

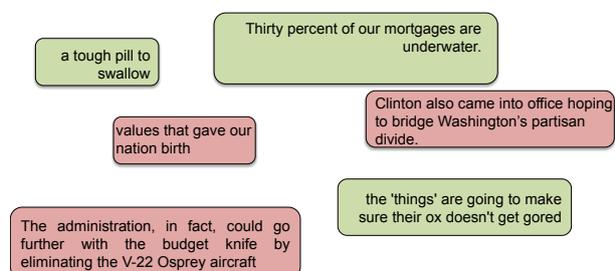


Figure 1: Polarity Classification

Figure 1 illustrates the polarity task in which the metaphors were classified into *Positive* or *Negative*. For instance, the metaphor “*tough pill to swallow*” has *Negative* polarity as it stands for something being hard to digest or comprehend, while the metaphor “*values that gave our nation birth*” has a *Positive* polarity as giving birth is like starting a new beginning.

### 4.2 Classification Algorithms

We model the metaphor polarity task as a classification problem in which, for a given collection of  $N$  training examples, where  $m_i$  is a metaphor and  $c_i$  is the polarity of  $m_i$ , the objective is to learn a classification function  $f : m_i \rightarrow c_i$  in which 1 stands for positive polarity and 0 stands for negative polarity. We tested five different machine learning algorithms such as Naive Bayes, SVM with polynomial kernel, SVM with RBF kernel, AdaBoost and Stacking, out of which AdaBoost performed the best. In our experimental study, we use the freely available implementations in Weka (Witten and Frank, 2005).

**Evaluation Measures:** To evaluate the goodness of the polarity classification algorithms, we calculate the f-score and accuracy on 10-fold cross validation.

### 4.3 Data Annotation

To conduct our experimental study, we have used annotated data provided by the Language Computer Corporation (LCC)<sup>1</sup>, which developed anno-

<sup>1</sup><http://www.languagecomputer.com/>

tation toolkit specifically for the task of metaphor detection, interpretation and affect assignment. They hired annotators to collect and annotate data for the English, Spanish, Russian and Farsi languages. The domain for which the metaphors were collected was *Governance*. It encompasses electoral politics, the setting of economic policy, the creation, application and enforcement of rules and laws. The metaphors were collected from political speeches, political websites, online newspapers among others (Mohler et al., 2013).

The annotation toolkit allowed annotators to provide for each metaphor the following information: the metaphor, the context in which the metaphor was found, the meaning of the metaphor in the source and target domains from the perspective of a native speaker. For example, in the **Context:** *And to all nations, we will speak for the values that gave our nation birth.*; the annotators tagged the **Metaphor:** *values that gave our nation birth*; and listed as **Source:** *mother gave birth to baby*; and **Target:** *values of freedom and equality motivated the creation of America*. The same annotators also provided the affect associated with the metaphor. The agreements of the annotators as measured by LCC are: .83, .87, .80 and .61 for the English, Spanish, Russian and Farsi languages.

In our study, the maximum length of a metaphor is a sentence, but typically it has the span of a phrase. The maximum length of a context is three sentences before and after the metaphor, but typically it has the span of one sentence before and after. In our study, the source and target domains are provided by the human annotators who agree on these definitions, however the source and target can be also automatically generated by an interpretation system or a concept mapper. The generation of source and target information is beyond the scope of this paper, but studying their impact on affect is important. At the same time, we want to show that if the technology for source/target detection and interpretation is not yet available, then how far can one reach by using the metaphor itself and the context around it. Later depending on the availability of the information sources and toolkits one can decide whether to integrate such information or to ignore it. In the experimental sections, we show how the individual information sources and their combination affects the resolution of the metaphor polarity and valence prediction tasks.

Table 1 shows the positive and negative class

distribution for each one of the four languages.

	Negative	Positive
ENGLISH	2086	1443
SPANISH	196	434
RUSSIAN	468	418
FARSI	384	252

Table 1: Polarity Class Distribution for Four Languages

The majority of the the annotated examples are for English. However, given the difficulty of finding bilingual speakers, we still managed to collect around 600 examples for Spanish and Farsi, and 886 examples for Russian.

#### 4.4 N-gram Evaluation and Results

N-gram features are widely used in a variety of classification tasks, therefore we also use them in our polarity classification task. We studied the influence of unigrams, bigrams and a combination of the two, and saw that the best performing feature set consists of the combination of unigrams and bigrams. In this paper, we will refer from now on to n-grams as the combination of unigrams and bigrams.

Figure 2 shows a study of the influence of the different information sources and their combination with n-gram features for English.

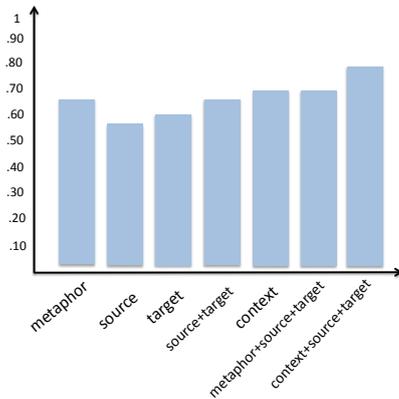


Figure 2: Influence of Information Sources for Metaphor Polarity Classification of English Texts

For each information source (metaphor, context, source, target and their combinations), we built a separate n-gram feature set and model, which was evaluated on 10-fold cross validation. The results from this study show that for English, the more information sources one combines, the higher the classification accuracy becomes.

Table 2 shows the influence of the information sources for Spanish, Russian and Farsi with the n-gram features. The best f-scores for each language are shown in bold. For Farsi and Russian high performances are obtained both with the context and with the combination of the context, source and target information. While for Spanish they reach similar performance.

	SPANISH	RUSSIAN	FARSI
Metaphor	71.6	71.0	62.4
Source	67.1	62.4	55.4
Target	68.9	67.2	62.4
Context	73.5	<b>77.1</b>	<b>67.4</b>
S+T	<b>76.6</b>	68.7	62.4
M+S+T	76.0	75.4	64.2
C+S+T	<b>76.5</b>	<b>76.5</b>	<b>68.4</b>

Table 2: N-gram features, F-scores on 10-fold validation for Spanish, Russian and Farsi

#### 4.5 LIWC as a Proxy for Metaphor Polarity

**LIWC Repository:** In addition to the n-gram features, we also used the Linguistic Inquiry and Word Count (LIWC) repository (Tausczik and Pennebaker, 2010), which has 64 word categories corresponding to different classes like emotional states, psychological processes, personal concerns among other. Each category contains a list of words characterizing it. For instance, the LIWC category *discrepancy* contains words like *should*, *could* among others, while the LIWC category *inhibition* contains words like *block*, *stop*, *constrain*. Previously LIWC was successfully used to analyze the emotional state of bloggers and tweeters (Quercia et al., 2011) and to identify deception and sarcasm in texts (Ott et al., 2011; González-Ibáñez et al., 2011). When LIWC analyzes texts it generates statistics like number of words found in category  $C_i$  divided by the total number of words in the text. For our metaphor polarity task, we use LIWC’s statistics of all 64 categories and feed this information as features for the machine learning classifiers. LIWC repository contains conceptual categories (dictionaries) both for the English and Spanish languages.

**LIWC Evaluation and Results:** In our experiments LIWC is applied to English and Spanish metaphor-rich texts since the LIWC category dictionaries are available for both languages. Table 3 shows the obtained accuracy and f-score results in English and Spanish for each one of the information sources.

	ENGLISH		SPANISH	
	Acc	Fscore	Acc	Fscore
Metaphor	<b>98.8</b>	<b>98.8</b>	87.9	87.2
Source	<b>98.6</b>	<b>98.6</b>	<b>97.3</b>	<b>97.3</b>
Target	<b>98.2</b>	<b>98.2</b>	<b>97.9</b>	<b>97.9</b>
Context	91.4	91.4	<b>93.3</b>	<b>93.2</b>
S+T	98.0	98.0	76.3	75.5
M+S+T	95.8	95.7	86.8	86.0
C+S+T	87.9	88.0	79.2	78.5

Table 3: LIWC features, Accuracy and F-scores on 10-fold validation for English and Spanish

The best performances are reached with individual information sources like metaphor, context, source or target instead of their combinations. The classifiers obtain similar performance for both languages.

**LIWC Category Relevance to Metaphor Polarity:** We also study the importance and relevance of the LIWC categories for the metaphor polarity task. We use information gain (IG) to measure the amount of information in bits about the polarity class prediction, if the only information available is the presence of a given LIWC category (feature) and the corresponding polarity class distribution. IG measures the expected reduction in entropy (uncertainty associated with a random feature) (Mitchell, 1997).

Figure 3 illustrates how certain categories occur more with the positive (in red color) vs negative (in green color) class. With the positive metaphors we observe the LIWC categories for present tense, social, affect and family, while for the negative metaphors we see LIWC categories for past tense, inhibition and anger.

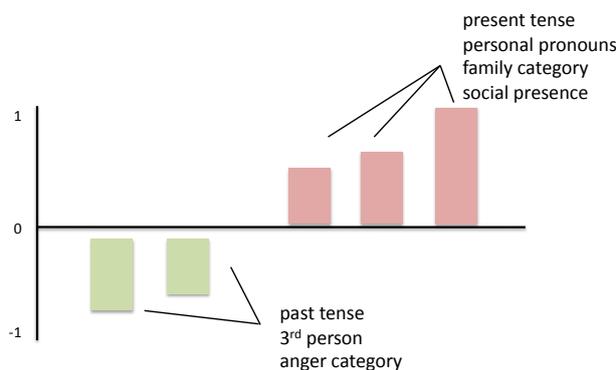


Figure 3: LIWC category relevance to Metaphor Polarity

In addition, we show in Figure 4 examples of the top LIWC categories according to IG ranking

for each one of the information sources.

Metaphor	Context	Source	Target
I	I		conj.
conj.	you		anger
anger		hate kill annoyed	affect
discrepancy		home	work
swear words	should could would	swear words	
inhibition	friend	religion	
body	affect	space	
relativity	sad		
home	inhibition	block constrain stop	
ingest	ingest		
work	work		

Figure 4: Example of LIWC Categories and Words

For metaphor texts, these categories are *I*, *conjunction*, *anger*, *discrepancy*, *swear words* among others; for contexts the categories are pronouns like *I*, *you*, *past tense*, *friends*, *affect* and so on. Our study shows that some of the LIWC categories are important across all information sources, but overall different triggers activate depending on the information source and the length of the text used.

#### 4.6 Comparative study

Figure 5 shows a comparison of the accuracy of our best performing approach for each language. For English and Spanish these are the LIWC models, while for Russian and Farsi these are the n-gram models. We compare the performance of the algorithms with a majority baseline, which assigns the majority class to each example. For instance, in English there are 3529 annotated examples, of which 2086 are positive and 1443 are negative. Since the positive class is the predominant one for this language and dataset, a majority classifier would have .59 accuracy in returning the positive class as an answer. Similarly, we compute the majority baseline for the rest of the languages.

	Accuracy	Majority Baseline	Difference
English	98.80	59.11	+39.69
Spanish	97.90	68.88	+29.02
Russian	77.00	52.82	+24.18
Farsi	72.20	60.30	+11.90

Figure 5: Best Accuracy Model and Comparison against a Majority Baseline for Metaphor Polarity Classification

As we can see from Figure 5 that all classifiers significantly outperform the majority base-

line. For Farsi the increment is +11.90, while for English the increment is +39.69. This means that the built classifiers perform much better than a random classifier.

#### 4.7 Lessons Learned

To summarize, in this section we have defined the task of polarity classification and we have presented a machine learning solution. We have used different feature sets and information sources to solve the task. We have conducted exhaustive evaluations for four different languages namely English, Spanish, Russian and Farsi. The learned lessons from this study are: (1) for n-gram usage, the larger the context of the metaphor, the better the classification accuracy becomes; (2) if present source and target information can further boost the performance of the classifiers; (3) LIWC is a useful resource for polarity identification in metaphor-rich texts; (4) analyzing the usages of tense like past vs. present and pronouns are important triggers for positive and negative polarity of metaphors; (5) some categories like *family*, *social presence* indicate positive polarity, while others like *inhibition*, *anger* and *swear words* are indicative of negative affect; (6) the built models significantly outperform majority baselines.

### 5 Task B: Valence Prediction

#### 5.1 Problem Formulation

**Task Definition:** Given metaphor-rich texts annotated with valence score (from  $-3$  to  $+3$ ), where  $-3$  indicates strong negativity,  $+3$  indicates strong positivity,  $0$  indicates neutral, the goal is to build a model that can predict without human supervision the valence scores of new previously unseen metaphors.

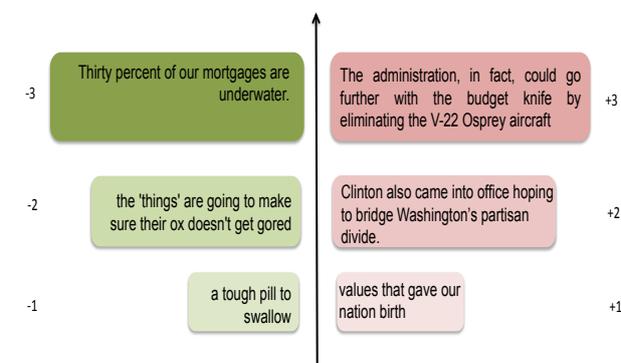


Figure 6: Valence Prediction

Figure 6 shows an example of the valence prediction task in which the metaphor-rich texts must be arranged by the intensity of the emotional state provoked by the texts. For instance,  $-3$  corresponds to very strong negativity,  $-2$  strong negativity,  $-1$  weak negativity (similarly for the positive classes). In this task we also consider metaphors with neutral affect. They are annotated with the  $0$  label and the prediction model should be able to predict such intensity as well. For instance, the metaphor “*values that gave our nation birth*”, is considered by American people that giving birth sets new beginning and has a positive score  $+1$ , but “*budget knife*” is more positive  $+3$  since tax cut is more important. As any sentiment analysis task, affect assignment of metaphors is also a subjective task and the produced annotations express the values, beliefs and understanding of the annotators.

#### 5.2 Regression Model

We model the valence task a regression problem, in which for a given metaphor  $m$ , we seek to predict the valence  $v$  of  $m$ . We do this via a parametrized function  $f: \hat{v} = f(m; w)$ , where  $w \in R^d$  are the weights. The objective is to learn  $w$  from a collection of  $N$  training examples  $\{ \langle m_i, v_i \rangle \}_{i=1}^N$ , where  $m_i$  are the metaphor examples and  $v_i \in R$  is the valence score of  $m_i$ .

Support vector regression (Drucker et al., 1996) is a well-known method for training a regression model by solving the following optimization problem:

$$\min_{w \in R^s} \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{i=1}^N \underbrace{\max(0, |v_i - f(m_i; w)| - \epsilon)}_{\epsilon\text{-insensitive loss function}}$$

where  $C$  is a regularization constant and  $\epsilon$  controls the training error. The training algorithm finds weights  $w$  that define a function  $f$  minimizing the empirical risk. Let  $h$  be a function from seeds into some vector-space representation  $\subseteq R^d$ , then the function  $f$  takes the form:  $f(m; w) = h(m)^T w = \sum_{i=1}^N \alpha_i K(m, m_i)$ , where  $f$  is re-parameterized in terms of a polynomial kernel function  $K$  with dual weights  $\alpha_i$ .  $K$  measures the similarity between two metaphoric texts. Full details of the regression model and its implementation are beyond the scope of this paper; for more details see (Schölkopf and Smola, 2001; Smola et al., 2003). In our experimental study, we use the freely available implementation of SVM in Weka (Witten and Frank, 2005).

**Evaluation Measures:** To evaluate the quality of the valence prediction model, we compare the actual valence score of the metaphor given by human annotators denoted with  $y$  against those valence scores predicted by the regression model denoted with  $x$ . We estimate the goodness of the regression model calculating both the correlation coefficient  $cc_{x,y} = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n \sum x_i^2 - (\sum x_i)^2} \sqrt{n \sum y_i^2 - (\sum y_i)^2}}$  and the mean squared error  $mse_{x,y} = \frac{\sum_{i=1}^n (x_i - \hat{x})^2}{n}$ . The two evaluation measures should be interpreted in the following manner. Intuitively the higher the correlation score is, the better the correlation between the actual and the predicted valence scores will be. Similarly the smaller the mean squared error rate, the better the regression model fits the valence predictions to the actual score.

### 5.3 Data Annotation

To conduct our valence prediction study, we used the same human annotators from the polarity classification task for each one of the English, Spanish, Russian and Farsi languages. We asked the annotators to map each metaphor on a  $[-3, +3]$  scale depending on the intensity of the affect associated with the metaphor.

Table 4 shows the distribution (number of examples) for each valence class and for each language.

	-3	-2	-1	0	+1	+2	+3
ENGLISH	1057	817	212	582	157	746	540
SPANISH	106	65	27	17	40	132	262
RUSSIAN	118	42	308	13	202	149	67
FARSI	147	117	120	49	91	63	98

Table 4: Valence Score Distribution for Each Language

### 5.4 Empirical Evaluation and Results

For each language and information source we built separate valence prediction regression models. We used the same features for the regression task as we have used in the classification task. Those include n-grams (unigrams, bigrams and combination of the two), LIWC scores. Table 5 shows the obtained correlation coefficient (CC) and mean squared error (MSE) results for each one of the four languages (English, Spanish, Russian and Farsi) using the dataset described in Table 4.

The Farsi and Russian regression models are based only on n-gram features, while the English and Spanish regression models have both n-gram and LIWC features. Overall, the CC for English

and Spanish is higher when LIWC features are used. This means that the LIWC based valence regression model approximates the predicted values better to those of the human annotators. The better valence prediction happens when the metaphor itself is used by LIWC. The MSE for English and Spanish is the lowest, meaning that the prediction is the closest to those of the human annotators. In Russian and Farsi the lowest MSE is when the combined metaphor, source and target information sources are used. For English and Spanish the smallest MSE or so called prediction error is 1.52 and 1.30 respectively, while for Russian and Farsi is 1.62 and 2.13 respectively.

### 5.5 Lessons Learned

To summarize, in this section we have defined the task of valence prediction of metaphor-rich texts and we have described a regression model for its solution. We have studied different feature sets and information sources to solve the task. We have conducted exhaustive evaluations in all four languages namely English, Spanish, Russian and Farsi. The learned lessons from this study are: (1) valence prediction is a much harder task than polarity classification both for human annotation and for the machine learning algorithms; (2) the obtained results showed that despite its difficulty this is still a plausible problem; (3) similarly to the polarity classification task, valence prediction with LIWC is improved when shorter contexts (the metaphor/source/target information source) are considered.

## 6 Conclusion

People use metaphor-rich language to express affect and often affect is expressed through the usage of metaphors. Therefore, understanding that the metaphor “*I was **boiling inside** when I saw him.*” has *Negative* polarity as it conveys feeling of anger is very important for interpersonal or multicultural communications.

In this paper, we have introduced a novel corpus of metaphor-rich texts for the English, Spanish, Russian and Farsi languages, which was manually annotated with the polarity and valence scores of the affect conveyed by the metaphors. We have studied the impact of different information sources such as the metaphor in isolation, the context in which the metaphor was used, the source and target domain meanings of the metaphor and

	RUSSIAN N-gram		FARSI N-gram		ENGLISH N-gram		SPANISH N-gram		ENGLISH LIWC		SPANISH LIWC	
	CC	MSE	CC	MSE	CC	MSE	CC	MSE	CC	MSE	CC	MSE
Metaphor	.45	1.71	.25	2.25	.36	2.50	.37	2.54	.74	1.52	.87	1.20
Source	.22	1.89	.11	2.42	.40	2.27	.22	2.43	.81	1.30	.85	1.28
Target	.25	1.91	.15	2.47	.37	2.41	.32	2.36	.72	1.56	.85	1.29
Context	.43	1.83	.32	2.38	.37	2.59	.40	2.37	.40	2.16	.67	1.92
S+T	.29	1.83	.18	2.38	.40	2.40	.41	2.19	.70	1.60	.78	1.53
M+S+T	.45	1.62	.29	2.13	.43	2.34	.43	2.14	.67	1.67	.78	1.53
C+S+T	.42	1.85	.26	2.61	.43	2.52	.39	2.41	.44	2.08	.64	1.96

Table 5: Valence Prediction, Correlation Coefficient and Mean Squared Error for English, Spanish, Russian and Farsi

their combination in order to understand how such information helps and impacts the interpretation of the affect associated with the metaphor. We have conducted exhaustive evaluation with multiple machine learning classifiers and different features sets spanning from lexical information to psychological categories developed by (Tausczik and Pennebaker, 2010). Through experiments carried out on the developed datasets, we showed that the proposed polarity classification and valence regression models significantly improve baselines (from 11.90% to 39.69% depending on the language) and work well for all four languages. From the two tasks, the valence prediction problem was more challenging both for the human annotators and the automated system. The mean squared error in valence prediction in the range  $[-3, +3]$ , where  $-3$  indicates strong negative and  $+3$  indicates strong positive affect for English, Spanish and Russian was around 1.5, while for Farsi was around 2.

The current findings and learned lessons reflect the properties of the collected data and its annotations. In the future we are interested in studying the affect of metaphors for domains different than *Governance*. We want to conduct studies with the help of social sciences who would research whether the tagging of affect in metaphors depends on the political affiliation, age, gender or culture of the annotators. Not on a last place, we would like to improve the built valence prediction models and to collect more data for Spanish, Russian and Farsi.

## Acknowledgments

The author would like to thank the reviewers for their helpful comments as well as the LCC annotators who have prepared the data and made this work possible. This research is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-

12-C-0025. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

## References

- Max Black. 1962. *Models and Metaphors*.
- Isabelle Blanchette, Kevin Dunbar, John Hummel, and Richard Marsh. 2001. Analogy use in naturalistic settings: The influence of audience, emotion and goals. *Memory and Cognition*, pages 730–735.
- Eric Breck, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of the 20th international joint conference on Artificial intelligence, IJCAI'07*, pages 2683–2688. Morgan Kaufmann Publishers Inc.
- Yejin Choi and Claire Cardie. 2009. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 590–598.
- Elizabeth Cawdord. 2009. Conceptual metaphors of affect. *Emotion Review*, pages 129–139.
- Harris Drucker, Chris J.C. Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. 1996. Support vector regression machines. In *Advances in NIPS*, pages 155–161.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentimentnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06)*, pages 417–422.
- Dedre Gentner. 1983. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170.

- Roberto González-Ibáñez, Smaranda Muresa n, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 581–586.
- Alistair Kennedy and Diana Inkpen. 2005. Sentiment classification of movie and product reviews using contextual valence shifters. *Computational Intelligence*, pages 110–125.
- Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04.
- George Lakoff and Mark Johnson. 1980. *Metaphors We Live By*. University of Chicago Press, Chicago.
- James H. Martin. 1988. Representing regularities in the metaphoric lexicon. In *Proceedings of the 12th conference on Computational linguistics - Volume 1*, COLING '88, pages 396–401.
- Thomas M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, Inc., 1 edition.
- Michael Mohler, David Bracewell, David Hinote, and Marc Tomlinson. 2013. Semantic signatures for example-based linguistic metaphor detection. In *The Proceedings of the First Workshop on Metaphor in NLP, (NAACL)*, pages 46–54.
- Yun Niu, Xiaodan Zhu, Jianhua Li, and Graeme Hirst. 2005. Analysis of polarity information in medical text. In *In: Proceedings of the American Medical Informatics Association 2005 Annual Symposium*, pages 570–574.
- Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 309–319.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.
- Livia Polanyi and Annie Zaenen. 2004. Contextual lexical valence shifters. In Yan Qu, James Shanahan, and Janyce Wiebe, editors, *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*. AAAI Press. AAAI technical report SS-04-07.
- Daniele Quercia, Jonathan Ellis, Licia Capra, and Jon Crowcroft. 2011. In the mood for being influential on twitter. In *the 3rd IEEE International Conference on Social Computing*.
- Antonio Reyes and Paolo Rosso. 2012. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decis. Support Syst.*, 53(4):754–760, November.
- Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Lang. Resour. Eval.*, 47(1):239–268, March.
- Bernhard Schölkopf and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. The MIT Press.
- Ekaterina Shutova and Simone Teufel. 2010. Metaphor corpus annotated for source - target domain mappings. In *International Conference on Language Resources and Evaluation*.
- Ekaterina Shutova, Lin Sun, and Anna Korhonen. 2010. Metaphor identification using verb and noun clustering. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1002–1010.
- Ekaterina Shutova. 2010a. Automatic metaphor interpretation as a paraphrasing task. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1029–1037.
- Ekaterina Shutova. 2010b. Models of metaphor in nlp. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 688–697.
- Catherine Smith, Tim Rumbell, John Barnden, Bob Hendley, Mark Lee, and Alan Wallington. 2007. Don't worry about metaphor: affect extraction for conversational agents. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 37–40. Association for Computational Linguistics.
- Alex J. Smola, Bernhard Schölkopf, and Bernhard Schölkopf. 2003. A tutorial on support vector regression. Technical report, Statistics and Computing.
- Carlo Strapparava and Rada Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 70–74. Association for Computational Linguistics, June.
- Yla R. Tausczik and James W. Pennebaker. 2010. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. *Journal of Language and Social Psychology*, 29(1):24–54, March.
- Marc T. Tomlinson and Bradley C. Love. 2006. From pigeons to humans: grounding relational learning in concrete examples. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, AAAI'06, pages 199–204. AAAI Press.
- Peter D. Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 417–424.

- Tony Veale and Guofu Li. 2012. Specifying viewpoint and information need with affective metaphors: a system demonstration of the metaphor magnet web app/service. In *Proceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 7–12.
- Tony Veale. 2012. A context-sensitive, multi-faceted model of lexico-conceptual affect. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 75–79.
- Janyce Wiebe and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. language resources and evaluation. In *Language Resources and Evaluation (formerly Computers and the Humanities)*.
- Yorick Wilks. 2007. A preferential, pattern-seeking, semantics for natural language inference. In *Words and Intelligence I*, volume 35 of *Text, Speech and Language Technology*, pages 83–102. Springer Netherlands.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354.
- Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, second edition.
- Ainur Yessenalina and Claire Cardie. 2011. Compositional matrix-space models for sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 172–182.

# Large tagset labeling using Feed Forward Neural Networks. Case study on Romanian Language

**Tiberiu Boros**

Research Institute for  
Artificial Intelligence “Mihai  
Drăgănescu”,  
Romanian Academy  
tibi@racai.ro

**Radu Ion**

Research Institute for  
Artificial Intelligence “Mihai  
Drăgănescu”,  
Romanian Academy  
radu@racai.ro

**Dan Tufiş**

Research Institute for  
Artificial Intelligence “Mihai  
Drăgănescu”,  
Romanian Academy  
tufis@racai.ro

## Abstract

Standard methods for part-of-speech tagging suffer from data sparseness when used on highly inflectional languages (which require large lexical tagset inventories). For this reason, a number of alternative methods have been proposed over the years. One of the most successful methods used for this task, called Tiered Tagging (Tufiş, 1999), exploits a reduced set of tags derived by removing several recoverable features from the lexicon morpho-syntactic descriptions. A second phase is aimed at recovering the full set of morpho-syntactic features. In this paper we present an alternative method to Tiered Tagging, based on local optimizations with Neural Networks and we show how, by properly encoding the input sequence in a general Neural Network architecture, we achieve results similar to the Tiered Tagging methodology, significantly faster and without requiring extensive linguistic knowledge as implied by the previously mentioned method.

## 1 Introduction

Part-of-speech tagging is a key process for various tasks such as information extraction, text-to-speech synthesis, word sense disambiguation and machine translation. It is also known as lexical ambiguity resolution and it represents the process of assigning a uniquely interpretable label to every word inside a sentence. The labels are called POS tags and the entire inventory of POS tags is called a tagset.

There are several approaches to part-of-speech tagging, such as Hidden Markov Models (HMM) (Brants, 2000), Maximum Entropy Classifiers (Berger et al., 1996; Ratnaparkhi, 1996), Bayesian Networks (Samuelsson, 1993), Neural

Networks (Marques and Lopes, 1996) and Conditional Random Fields (CRF) (Lafferty et al., 2001). All these methods are primarily intended for English, which uses a relatively small tagset inventory, compared to highly inflectional languages. For the later mentioned languages, the lexicon tagsets (called morpho-syntactic descriptions (Calzolari and Monachini, 1995) or MSDs) may be 10-20 times or even larger than the best known tagsets for English. For instance Czech MSD tagset requires more than 3000 labels (Collins et al., 1999), Slovene more than 2000 labels (Erjavec and Krek, 2008), and Romanian more than 1100 labels (Tufiş, 1999). The standard tagging methods, using such large tagsets, face serious data sparseness problems due to lack of statistical evidence, manifested by the non-robustness of the language models. When tagging new texts that are not in the same domain as the training data, the accuracy decreases significantly. Even tagging in-domain texts may not be satisfactorily accurate.

One of the most successful methods used for this task, called Tiered Tagging (Tufiş, 1999), exploits a reduced set of tags derived by removing several *recoverable* features from the lexicon morpho-syntactic descriptions. According to the MULTEXT EAST lexical specifications (Erjavec and Monachini, 1997), the Romanian tagset consists of a number of 614 MSD tags (by exploiting the case and gender regular syncretism) for wordforms and 10 punctuation tags (Tufiş et al., 1997), which is still significantly larger than the tagset of English. The MULTEXT EAST version 4 (Erjavec, 2010) contains specifications for a total of 16 languages: Bulgarian, Croatian, Czech, Estonian, English, Hungarian, Romanian,

Serbian, Slovene, the Resian dialect of Slovene, Macedonian, Persian, Polish, Russian, Slovak, and Ukrainian

The strategy of the Tiered Tagging methodology is to use a reduced tagset (called CTAG-set), where a CTAG is a generalization of a MSD, from which recoverable context-irrelevant features are removed. For instance, the attribute for gender (masculine ‘m’ or feminine ‘f’) from MSDs ‘Ncfsrn’ and ‘Ncmsrn’ is deleted to obtain the CTAG ‘NSRN’, because the gender information can be deterministically recovered based on the CTAG and the wordform itself. The recovering of the left out attributes (Figure 1) is based on the lexicons, linguistic rules and, in the case of unknown words, on ML techniques (Ceaşu, 2006). When tagging with CTAGs, one can use any statistical POS tagging method such as HMMs, Maximum Entropy Classifiers, Bayesian Networks, CRFs, etc., followed by the CTAG to MSD recovery.

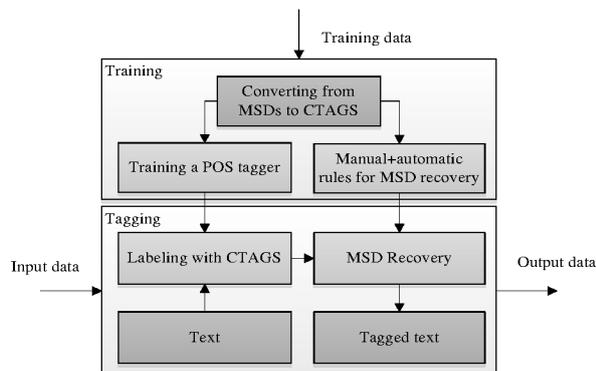


Figure 1 - Tiered Tagging methodology

The language dependent process of manually inferring linguistic rules for MSD recovery requires good knowledge of the target language and also extensive amounts of time invested in testing and re-design. It is difficult even for a native speaker to create such rules without in-depth linguistic knowledge.

In this article, we propose an alternative solution based on local optimizations with feed-forward neural networks. Our method eliminates the need for the two stage processing, is much faster at run time and is comparatively accurate with the Tiered Tagging implemented in the TTL tagger (Ion, 2007), available in the METASHARE Platform<sup>1</sup>.

<sup>1</sup> <http://ws.racai.ro:9191/>

## 2 Large tagset part-of-speech tagging with feed-forward neural networks

Although removing the recoverable attributes, as proposed in the Tiered Tagging approach, helps the goal of squeezing the lexical tagset to a reasonable size, valuable information for contextual disambiguation is also lost. For example, the gender agreement rule, valid in many languages, cannot be exploited unless the gender attribute is present in the tags.

Our proposal to deal with large tagsets without removing contextually useful information is based on Feed Forward Neural Networks (FFNN). FFNN are known for their simplicity and robustness in finding and recombining patterns inside data.

Several neural network architectures have been proposed for the task of part-of-speech tagging. Schmid (1994) proposed a FFNN architecture for part-of-speech tagging obtaining a 96.22% accuracy. In his paper, he argues that neural networks are preferable to other methods, when the training set is small. He compares his results with a HMM tagger (94.24%) and a trigram tagger (96.02%), both trained and tested on the same corpora as his FFNN tagger (the Penn-Treebank corpus). A similar approach is presented by Marques and Lopes (1996). In their paper, the authors come to similar conclusions as those presented in Schmid (1994). We support these findings with an additional argument, namely the better fit for managing large tagsets.

In both approaches mentioned before, the network is trained so that from the input vector, to output a real valued vector. Each value in the output vector is generated by a distinct neuron, and corresponds to a unique tag in the tagset (e.g. 100 tags means the network contains 100 neurons on the output layer). The input vector for predicting the tag of the current word encodes the tags for the previously tagged words and the probable tags for the current and following two words, estimated using Maximum Likelihood Estimation (MLE):

$$P(t|w) = \frac{C(w, t)}{C(w)} \quad (1)$$

- $P(t|w)$  - The probability of the word  $w$  having tag  $t$
- $C(w, t)$  - The total number of times, the word  $w$  appears with tag  $t$  in the training corpus
- $C(w)$  - The total number of times, the word  $w$  appears in the training corpus

In the case of out-of-vocabulary (OOV) words, both approaches use suffix analysis to determine the most probable tags that can be assigned to the current word.

To clarify how these two methods work, if we want to train the network to label the current word, using a context window of 1 (previous tag, current possible tags, and possible tags for the next word) and if we have, say 100 tags in the tagset, the input is a real valued vector of 300 sub-unit elements and the output is a vector which contains 100 elements, also sub-unit real numbers. As mentioned earlier, each value in the output vector corresponds to a distinct tag from tagset and the tag assigned to the current word is chosen to correspond to the maximum value inside the output vector.

The previously proposed methods still suffer from the same issue of data sparseness when applied to MSD tagging. However, in our approach, we overcome the problem through a different encoding of the input data (see section 2.1).

The power of neural networks results mainly from their ability to attain activation functions over different patterns via their learning algorithm. By properly encoding the input sequence, the network chooses which input features contribute in determining the output features for MSDs (e.g. patterns composed of part of speech, gender, case, type etc. contribute independently in selecting the optimal output sequence). This way, we removed the need for explicit MSD to CTAG conversion and MSD recovery from CTAGs.

## 2.1 The MSD binary encoding scheme

A MSD language independently encodes a part of speech (POS) with the associated lexical attribute values as a string of positional ordered character codes (Erjavec, 2004). The first character is an upper case character denoting the part of speech (e.g. 'N' for nouns, 'V' for verbs, 'A' for adjectives, etc.) and the following characters (lower letters or '-') specify the instantiations of the characteristic lexical attributes of the POS. For example, the MSD 'Ncfsrn', specifies a noun (the first character is 'N') the type of which is common ('c', the second character), feminine gender ('f'), singular number ('s'), in nominative/accusative case ('r') and indefinite form ('n'). If a specific attribute is not relevant for a language, or for a given combination of feature-values, the character '-' is used in the corresponding position. For a

language which does not morphologically mark the gender and definiteness features, the earlier exemplified MSD will be encoded as 'Nc-sr-'.

In order to derive a binary vector for each of the 614 MSDs of Romanian we proceeded to:

1. List and sort all possible POSes of Romanian (16 POSes) and form a binary vector with 16 positions in which position  $k$  is equal 1 only if the respective MSD has the corresponding POS (i.e. the  $k$ -th POS in the sorted list of POSes);
2. List and sort all possible values of all lexical attributes (disregarding the wildcard '-') for all POSes (94 values) and form another binary vector with 94 positions such that the  $k$ -th position of this vector is 1 if the respective MSD has an attribute with the corresponding value;
3. Concatenate the vectors from steps 1 and 2 and obtain the binary codification of a MSD as a 110-position binary vector.

## 2.2 The training and tagging procedure

The tagger automatically assigns four dummy tokens (two at the beginning and two at the end) to the target utterance and the neural network is trained to automatically assign a MSD given the context (two previously assigned tags and the possible tags for the current and following two words) of the current word (see below for details).

In our framework a training example consists of the features extracted for a single word inside an utterance as input and its MSD within that utterance as output. The features are extracted from a window of 5 words centered on the current word. A single word is characterized by a vector that encodes either its assigned MSD or its possible MSDs. To encode the possible MSDs we use equation 2, where each possible attribute  $a$ , has a single corresponding position inside the encoded vector.

$$P(a|w) = \frac{C(w, a)}{C(w)} \quad (2)$$

*Note that we changed the probability estimates to account for attributes not tags.*

To be precise, for every word  $w_k$ , we obtain its input features by concatenating a number of 5 vectors. The first two vectors encode the MSDs assigned to the previous two words ( $w_{k-1}$  and  $w_k$ ).

2). The next three vectors are used to encode the possible MSDs for the current word ( $w_k$ ) and the following two words ( $w_{k+1}$  and  $w_{k+2}$ ).

During training, we also compute a list of suffixes with associated MSDs, which is used at run-time to build the possible MSDs vector for unknown words. When such words are found within the test data, we approximate their possible MSDs vector using a variation of the method proposed by Brants (2000).

When the tagger is applied to a new utterance, the system iteratively calculates the output MSD for each individual word. Once a label has been assigned to a word, the word's associated vector is edited so it will have the value of 1 for each attribute present in its newly assigned MSD.

As a consequence of encoding each individual attribute separately for MSDs, the tagger can assign new tags (that were never associated with the current word in the training corpus). Although this is a nice behavior for dealing with unknown words it is often the case that it assigns attribute values that are not valid for the wordform. To overcome these types of errors we use an additional list of words with their allowed MSDs. For an OOV word, the list is computed as a union from all MSDs that appeared with the suffixes that apply to that word.

When the tagger has to assign a MSD to a given word, it selects one from the possible wordform's MSDs in its wordform/MSDs associated list using a simple distance function:

$$\min_{e \in P} \sum_{k=0}^n |o_k - e_k| \quad (3)$$

$P$  - The list of all possible MSDs for the given word  
 $n$  - The length of the MSD encoding (110 bits)  
 $o$  - The output of the Neural Network for the current word  
 $e$  - Binary encoding for a MSD in  $P$

### 3 Network hyperparameters

In our experiments, we used a fully connected, feed forward neural network with 3 layers (1 input layer, 1 hidden layer and 1 output layer)

and a sigmoid activation function (equation 3). While other network architectures such as recurrent neural networks may prove to be more suitable for this task, they are extremely hard to train, thus, we traded the advantages of such architectures for the robustness and simplicity of the feed-forward networks.

$$f(t) = \frac{1}{1 + e^{-t}} \quad (3)$$

$f(t)$  - Neuron output  
 $t$  - The weighted sum of all the neuron outputs from the previous layer

Based on the size of the vectors used for MSD encoding, the output layer has 110 neurons and the input layer is composed of 550 (5 x 110) neurons.

In order to fully characterize our system, we took into account the following parameters: accuracy, runtime speed, training speed, hidden layer configuration and the number of optimal training iterations. These parameters have complex dependencies and relations among each other. For example, the accuracy, the optimal number of training iterations, the training and the runtime speed are all highly dependent on the hidden layer configuration. Small hidden layer give high training and runtime speeds, but often under-fit the data. If the hidden layer is too large, it can easily over-fit the data and also has a negative impact on the training and runtime speed. The number of optimal training iterations changes with the size of the hidden layer (larger layers usually require more training iterations).

To obtain the trade-offs between the above mentioned parameters we devised a series of experiments, in all of which we used the "1984" MSD annotated corpus, which is composed of 118,025 words. We randomly kept out approximately 1/10 (11,960 words) of the training corpus for building a cross-validation set. The baseline accuracy on the cross-validation set (i.e. returning the most probable tag) is 93.29%. We also used an additional inflectional wordform/MSD lexicon composed of approximately 1 million hand-validated entries.

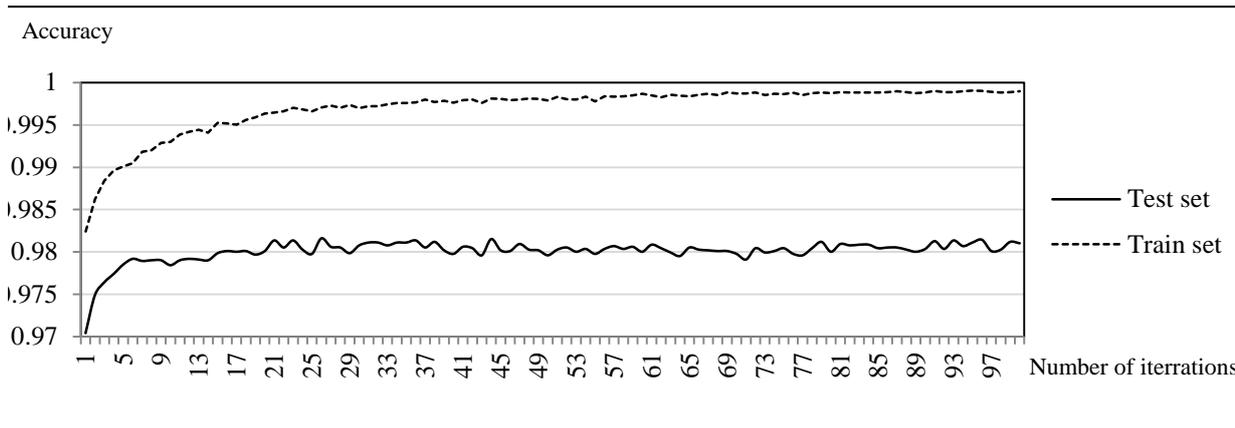


Figure 2 - 130 hidden layer network test and train set tagging accuracy as a function of the number of iterations

The first experiment was designed to determine the trade-off between the run-time speed and the size of the hidden layer. We made a series of experiments disregarding the tagging accuracy.

Hidden size	Time (ms)	Words/sec
50	1530	7816
70	1888	6334
90	2345	5100
110	2781	4300
130	3518	3399
150	5052	2367
170	5466	2188
190	6734	1776
210	7096	1685
230	8332	1435
250	9576	1248
270	10350	1155
290	11080	1079
310	12364	967

Table 1 - Execution time vs. number of neurons on the hidden layer

Because, for a given number of neurons in the hidden layer, the tagging speed is independent on the tagging accuracy, we partially trained (using one iteration and only 1000 training sentences) several network configurations. The first network only had 50 neurons in the hidden layer and for the next networks, we incremented the hidden layer size by 20 neurons until we reached 310 neurons. The total number of tested networks is 14. After this, we measured the time it took to tag the 1984 test corpus (11,960 words) for each individual network, as an average of 3 tagging runs in order to reduce the impact of the operating system load on the tagger (Table 1 shows the figures).

Determining the optimal size of the hidden layer is a very delicate subject and there are no perfect solutions, most of them being based on trial and error: small-sized hidden layers lead to under-fitting, while large hidden layers usually cause over-fitting. Also, because of the trade-off between runtime speed and the size of hidden layers, and if runtime speed is an important factor in a particular NLP application, then hidden layers with smaller number of neurons are preferable, as they surely do not over-fit the data and offer a noticeable speed boost.

hidden layer	Train set accuracy	Cross validation accuracy
50	99.18	97.95
70	99.20	98.02
90	99.27	98.03
110	99.29	98.05
130	99.35	98.12
150	99.35	98.09
170	99.41	98.07
190	99.40	98.10
210	99.40	98.21

Table 2 - Train and test accuracy rates for different hidden layer configurations

As shown in Table 1, the runtime speed of our system shows a constant decay when we increase the hidden layer size. The same decay can be seen in the training speed, only this time by an order of magnitude larger. Because training a single network takes a lot of time, this experiment was designed to estimate the size of the hidden layer which offers good performance in tagging. To do this, we individually trained a number of networks in 30 iterations, using various hidden layer configurations (50, 70, 90,

110, 130, 150, 170, 190, and 210 neurons) and 5 initial random initializations of the weights. For each configuration, we stored the accuracy of reproducing the learning data (the tagging of the training corpus) and the accuracy on the unseen data (test sets). The results are shown in Table 2. Although a hidden layer of 210 neurons did not seem to over-fit the data, we stopped the experiment, as the training time got significantly longer.

The next experiment was designed to see how the number of training iterations influences the tagging performance of networks with different hidden layer configurations. Intuitively, the training process must be stopped when the network begins to over-fit the data (i.e. the train set accuracy increases, but the test set accuracy drops). Our experiments indicate that this is not always the case, as in some situations the continuation of the training process leads to better results on the cross-validation data (as shown in Figure 2). So, the problem comes to determining which is the most stable configuration of the neural network (i.e. which hidden unit size will be most likely to return good results on the test set) and establish the number of iterations it takes for the system to be trained. To do this, we ran the training procedure for 100 iterations and for each training iteration, we computed the accuracy rate of every individual network on the cross-validation set (see Table 3 for the averaged values). As shown, the network configuration using 130 neurons on the hidden layer is most likely to produce better results on the cross-validation set regardless of the number of iterations.

Although, some other configurations provided better figures for the maximum accuracy, their average accuracy is lower than that of the 130 hidden unit network. Other good candidates are the 90 and 110 hidden unit networks, but not the larger valued ones, which display a lower average accuracy and also significantly slower tagging speeds.

The most suitable network configuration for a given task depends on the language, MSD encoding size, speed and accuracy requirements. In our own daily applications we use the 130 hidden unit network. After observing the behavior of the various networks on the cross-validation set we determined that a good choice is to stop the training procedure after 40 iterations.

Hidden units	Avg. acc.	Max. acc.	St. dev.
50	97.94	98.31	0.127002
70	98.03	98.31	0.12197
50	97.94	98.37	0.139762
70	98.03	98.43	0.124996
90	98.07	98.39	0.134487
110	98.08	98.45	0.127109
130	98.14	98.44	0.136072
150	98.01	98.36	0.143324
170	97.94	98.36	0.122834

Table 3 - Average and maximum accuracy for various hidden layer configuration calculated over 100 training iterations on the test set

To obtain the accuracy of the system, in our last experiment we used the 130 hidden unit network and we performed the training/testing procedure on the 1984 corpus, using 10-fold validation and 30 random initializations. The final accuracy was computed as an average between all the accuracy figures measured at the end of the training process (after 40 iterations). The first 1/10 of the 1984 corpus on which we tuned the hyperparameters was not included in the test data, but was used for training. The mean accuracy of the system (98.41%) was measured as an average of 270 values.

#### 4 Comparison to other methods

In his work, Ceașu (2006) presents a different approach to MSD tagging using the Maximum Entropy framework. He presents his results on the same corpus we used for training and testing (the 1984 corpus) and he compares his method (98.45% accuracy) with the Tiered Tagging methodology (97.50%) (Tufiş and Dragomirescu, 2004).

Our Neural Network approach obtained similar (slightly lower) results (98.41%), although it is arguable that our split/train procedure is not identical to the one used in his work (no details were given as how the 1/10 of the training corpus was selected). Also, our POS tagger detected cases where the annotation in the Gold Standard was erroneous. One such example is in “lame de ras” (English “razor blades”) where “lame” (English “blades”) is a noun, “de” (“for”) is a preposition and “ras” (“shaving”) is a supine verb (with a past participle form) which was incorrectly annotated as a noun.

## 5 Network pattern analysis

Using feed-forward neural networks gives the ability to outline what input features contribute to the selection of various MSD attribute values in the output layer which might help in reducing the tagset and thus, redesigning the network topology with beneficial effects both on the speed and accuracy.

To determine what input features contribute to the selection of certain MSD attribute values, one can analyze the weights inside the neural network and extract the input  $\rightarrow$  output links that are formed during training. We used the network with 130 units on the hidden layer, which was previously trained for 100 iterations. Based on the input encoding, we divided the features into 5 groups (one group for each MSD inside the local context – two previous MSDs, current and following two possible MSDs). For a target attribute value (noun, gender feminine, gender masculine, etc.) and for each input group, we selected the top 3 input values which support the decision of assigning the target value to the attribute (features that increase the output value) and the top 3 features which discourage this decision (features that decrease the output value). For clarity, we will use the following notations for the groups:

- $G_{-2}$ : group one – the assigned MSD for the word at position  $i-2$
- $G_{-1}$ : group two – the assigned MSD for the word at position  $i-1$
- $G_0$ : group three – the possible MSDs for the word at position  $i$
- $G_1$ : group four – the possible MSDs for the word at position  $i+1$
- $G_2$ : group five – the possible MSDs for the word at position  $i+2$

where  $i$  corresponds to the position of the word which is currently being tagged. Also, we classify the attribute values into two categories (C): (P) *want to see* (support the decision) and (N) *don't want to see* (discourage the decision).

Table 4 shows partial ( $G_{-1}$   $G_0$   $G_1$ ) examples of two target attribute values (cat=Noun and gender=Feminine) and their corresponding input features used for discrimination.

Target value	Group	C	Attribute values
Noun	$G_{-1}$	P	main (of a verb), <b>article</b> , masculine (gender of a noun/adjective)

		N	particle, conjunctive particle, auxiliary (of a verb), demonstrative (of a pronoun)
		$G_0$	P
	N		adverb, pronoun, numeral, interrogative/relative (of a pronoun)
	$G_1$	P	<b>genitive/dative (of a noun/adjective)</b> , particle, punctuation
N		conjunctive particle, strong (of a pronoun), non-definite (of a noun/adjective), exclamation mark	
Fem.	$G_{-1}$	P	main (of a verb), preposition, <b>feminine (of a noun/adjective)</b>
		N	auxiliary (of a verb), particle, demonstrative (of a pronoun)
	$G_0$	P	<b>feminine (of a noun/adjective)</b> , nominative/accusative (of a noun/adjective), past (of a verb)
		N	masculine (of a noun/adjective), auxiliary (of a verb), interrogative/relative (of a pronoun), adverb
	$G_1$	P	dative/genitive (of a noun/adjective), indicative (of a verb), <b>feminine (of a noun/adjective)</b>
		N	conjunctive particle, future particle, nominative/accusative (of a noun/adjective)

Table 4 – P/N features for various attribute values.

For instance, when deciding on whether to give a noun (N) label to current position ( $G_0$ ), we can see that the neural network has learned some interesting dependencies: at position  $G_{-1}$  we find an article (which frequently determines a noun) and at the current position it is very important for the word being tagged to actually be a common or proper noun (either by lexicon lookup or by suffix guessing) and not be an adverb, pronoun or numeral (POSeS that cannot be found in the typical ambiguity class of a noun). At the next position of the target ( $G_1$ ) we also find a noun in genitive or dative, corresponding to a frequent construction in Romanian, e.g. “mașina băiatului” being a sequence of two nouns, the second at genitive/dative.

If the neural network outputs the feminine gender to its current MSD, one may see that it

has actually learned the agreement rules (at least locally): the feminine gender is present both before ( $G_{-1}$ ) the target word as well as after it ( $G_1$ ).

## 6 Conclusions and future work

We presented a new approach for large tagset part-of-speech tagging using neural networks. An advantage of using this methodology is that it does not require extensive knowledge about the grammar of the target language. When building a new MSD tagger for a new language one is only required to provide the training data and create an appropriate MSD encoding system and as shown, the MSD encoding algorithm is fairly simple and our proposed version works for any other MSD compatible encoding, regardless of the language.

Observing which features do not participate in any decision helps design custom topologies for the Neural Network, and provides enhancements in both speed and accuracy. The configurable nature of our system allows users to provide their own MSD encodings, which permits them to mask certain features that are not useful for a given NLP application.

If one wants to process a large amount of text and is interested only in assigning grammatical categories to words, he can use a MSD encoding in which he strips off all unnecessary features. Thus, the number of necessary neurons would decrease, which assures faster training and tagging. This is of course possible in any other tagging approaches, but our framework supports this by masking attributes inside the MSD encoding configuration file, without having to change anything else in the training corpus. During testing the system only verifies if the MSD encodings are identical and the displayed accuracy directly reflects the performance of the system on the simplified tagging schema.

We also proposed a methodology for selecting a network configurations (i.e. number of hidden units), which best suites the application requirements. In our daily applications we use a network with 130 hidden units, as it provides an optimal speed/accuracy trade-off (approx. 3400 words per second with very good average accuracy).

The tagger is implemented as part of a larger application that is primarily intended for text-to-speech (TTS) synthesis. The system is free for non-commercial use and we provide both web and desktop user-interfaces. It is part of the

METASHARE platform and available online<sup>2</sup>. Our primary goal was to keep the system language independent, thus all our design choices are based on the necessity to avoid using language specific knowledge, when possible. The application supports various NLP related tasks such as lexical stress prediction, syllabification, letter-to-sound conversion, lemmatization, diacritic restoration, prosody prediction from text and the speech synthesizer uses unit-selection.

From the tagging perspective, our future plans include testing the system on other highly inflectional languages such as Czech and Slovene and investigating different methods for automatically determining a more suitable custom network topology, such as genetic algorithms.

## Acknowledgments

The work reported here was funded by the project METANET4U by the European Commission under the Grant Agreement No 270893

---

<sup>2</sup> <http://ws.racai.ro:9191>

## References

- Berger, A. L., Pietra, V. J. D. and Pietra, S. A. D. 1996. *A maximum entropy approach to natural language processing*. Computational linguistics, 22(1), 39-71.
- Brants, T. 2000. *TnT: a statistical part-of-speech tagger*. In Proceedings of the sixth conference on applied natural language processing (pp. 224-231). Association for Computational Linguistics.
- Calzolari, N. and Monachini M. (eds.). 1995. *Common Specifications and Notation for Lexicon Encoding and Preliminary Proposal for the Tagsets*. MULTEXT Report, March.
- Ceașu, A. 2006. *Maximum entropy tiered tagging*. In Proceedings of the 11th ESSLLI Student Session (pp. 173-179).
- Collins, M., Ramshaw, L., Hajič, J. and Tillmann, C. 1999. *A statistical parser for Czech*. In Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics (pp. 505-512). Association for Computational Linguistics.
- Erjavec, T. and Monachini, M. (Eds.). 1997. *Specifications and Notation for Lexicon Encoding*. Deliverable D1.1 F. Multext-East Project COP-106.
- Erjavec, T. 2004. *MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora*. In Fourth International Conference on Language Resources and Evaluation, LREC (Vol. 4, pp. 1535-1538).
- Erjavec, T. and Krek, S. 2008. *The JOS morphosyntactically tagged corpus of Slovene*. In Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC'08.
- Erjavec, T. 2010. *MULTEXT-East Version 4: Multilingual Morphosyntactic Specifications, Lexicons and Corpora*. In Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), Valletta, Malta. European Language Resources Association (ELRA) ISBN 2-9517408-6-7.
- Lafferty, J., McCallum, A. and Pereira, F. C. 2001. *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*.
- Marques, N. C. and Lopes, G. P. 1996. *A neural network approach to part-of-speech tagging*. In Proceedings of the 2nd Meeting for Computational Processing of Spoken and Written Portuguese (pp. 21-22).
- Ratnaparkhi, A. 1996. *A maximum entropy model for part-of-speech tagging*. In Proceedings of the conference on empirical methods in natural language processing (Vol. 1, pp. 133-142).
- Samuelsson, C. 1993. *Morphological tagging based entirely on Bayesian inference*. In 9th Nordic Conference on Computational Linguistics.
- Schmid, H. 1994. *Part-of-speech tagging with neural networks*. In Proceedings of the 15th conference on Computational linguistics-Volume 1 (pp. 172-176). Association for Computational Linguistics.
- Tufiș, D., Barbu A.M., Pătrașcu V., Rotariu G. and Popescu C. 1997. *Corpora and Corpus-Based Morpho-Lexical Processing*. In Recent Advances in Romanian Language Technology, (pp. 35-56). Romanian Academy Publishing House, ISBN 973-27-0626-0.
- Tufiș, D. 1999. *Tiered tagging and combined language models classifiers*. In Text, Speech and Dialogue (pp. 843-843). Springer Berlin/Heidelberg.
- Tufiș, D., and Dragomirescu, L. 2004. *Tiered tagging revisited*. In Proceedings of the 4th LREC Conference (pp. 39-42).

# Learning to lemmatise Polish noun phrases

Adam Radziszewski

Institute of Informatics, Wrocław University of Technology

Wybrzeże Wyspiańskiego 27

Wrocław, Poland

adam.radziszewski@pwr.wroc.pl

## Abstract

We present a novel approach to noun phrase lemmatisation where the main phase is cast as a tagging problem. The idea draws on the observation that the lemmatisation of almost all Polish noun phrases may be decomposed into transformation of singular words (tokens) that make up each phrase. We perform evaluation, which shows results similar to those obtained earlier by a rule-based system, while our approach allows to separate chunking from lemmatisation.

## 1 Introduction

Lemmatisation of word forms is the task of finding base forms (lemmas) for each token in running text. Typically, it is performed along POS tagging and is considered crucial for many NLP applications. Similar task may be defined for whole noun phrases (Degórski, 2011). By lemmatisation of noun phrases (NPs) we will understand assigning each NP a grammatically correct NP corresponding to the same phrase that could stand as a dictionary entry.

The task of NP lemmatisation is rarely considered, although it carries great practical value. For instance, any keyword extraction system that works for a morphologically rich language must deal with lemmatisation of NPs. This is because keywords are often longer phrases (Turney, 2000), while the user would be confused to see inflected forms as system output. Similar situation happens when attempting at terminology extraction from domain corpora: it is usually assumed that domain terms are subclass of NPs (Marciniak and Mykowiecka, 2013).

In (1) we give an example Polish noun phrase (*‘the main city of the municipality’*). Throughout the paper we assume the usage of the tagset

of the National Corpus of Polish (Przepiórkowski, 2009), henceforth called NCP in short. The orthographic form (1a) appears in instrumental case, singular. Phrase lemma is given as (1b). Lemmatisation of this phrase consists in reverting case value of the main noun (*miasto*) as well as its adjective modifier (*główne*) to nominative (nom). Each form in the example is in singular number (sg), *miasto* has neuter gender (n), *gmina* is feminine (f).

- (1) a. *głównym miastem gminy*  
main city municipality  
inst:sg:n inst:sg:n gen:sg:f
- b. *główne miasto gminy*  
main city municipality  
nom:sg:n nom:sg:n gen:sg:f

According to the lemmatisation principles accompanying the NCP tagset, adjectives are lemmatised as masculine forms (*główny*), hence it is not sufficient to take word-level lemma nor the orthographic form to obtain phrase lemmatisation. Degórski (2011) discusses some similar cases. He also notes that this is not an easy task and lemma of a whole NP is rarely a concatenation of lemmas of phrase components. It is worth stressing that even the task of word-level lemmatisation is non-trivial for inflectional languages due to a large number of inflected forms and even larger number of syncretisms. According to Przepiórkowski (2007), “a typical Polish adjective may have 11 textually different forms (...) but as many as 70 different tags (2 numbers  $\times$  7 cases  $\times$  5 genders)”, which indicates the scale of the problem. What is more, several syntactic phenomena typical for Polish complicate NP lemmatisation further. E.g., adjectives may both precede and follow nouns they modify; many English prepositional phrases are realised in Polish using oblique case without any proposition (e.g., there is no standard Polish coun-

terpart for the preposition *of* as genitive case is used for this purpose).

In this paper we present a novel approach to noun phrase lemmatisation where the main phase is cast as a tagging problem and tackled using a method devised for such problems, namely Conditional Random Fields (CRF).

## 2 Related works

NP lemmatisation received very little attention. This situation may be attributed to prevalence of works targeted at English, where the problem is next to trivial due to weak inflection in the language.

The only work that contains a complete description and evaluation of an approach to this task we were able to find is the work of Degórski (2011). The approach consists in incorporating phrase lemmatisation rules into a shallow grammar developed for Polish. This is implemented by extending the Spejdl shallow parsing framework (Buczyński and Przepiórkowski, 2009) with a rule action that is able to generate phrase lemmas. Degórski assumes that lemma of each NP may be obtained by concatenating each token’s orthographic form, lemma or ‘half-lemmatised’ form (e.g. grammatical case normalised to nominative, while leaving feminine gender). The other assumption is to neglect letter case: all phrases are converted to lower case and this is not penalised during evaluation. For development and evaluation, two subsets of NCP were chosen and manually annotated with NP lemmas: development set (112 phrases) and evaluation set (224 phrases). Degórski notes that the selection was not entirely random: two types of NPs were deliberately omitted, namely foreign names and “a few groups for which the proper lemmatisation seemed very unclear”. The final evaluation was performed in two ways. First, it is shown that the output of the entire system intersects only with 58.5% of the test set. The high error rate is attributed to problems with identifying NP boundaries correctly (29.5% of test set was not recognised correctly with respect to phrase boundaries). The other experiment was to limit the evaluation to those NPs whose boundaries were recognised correctly by the grammar (70.5%). This resulted in 82.9% success rate.

The task of phrase lemmatisation bears a close resemblance to a more popular task, namely lemmatisation of named entities. Depending on the

type of named entities considered, those two may be solved using similar or significantly different methodologies. One approach, which is especially suitable for person names, assumes that nominative forms may be found in the same source as the inflected forms. Hence, the main challenge is to define a similarity metric between named entities (Piskorski et al., 2009; Kocoń and Piasecki, 2012), which can be used to match different mentions of the same names. Other named entity types may be realised as arbitrary noun phrases. This calls for more robust lemmatisation strategies.

Piskorski (2005) handles the problem of lemmatisation of Polish named entities of various types by combining specialised gazetteers with lemmatisation rules added to a hand-written grammar. As he notes, organisation names are often built of noun phrases, hence it is important to understand their internal structure. Another interesting observation is that such organisation names are often structurally ambiguous, which is exemplified with the phrase (2a), being a string of items in genitive case (*‘of the main library of the Higher School of Economics’*). Such cases are easier to solve when having access to a collocation dictionary — it may be inferred that there are two collocations here: *Biblioteka Główna* and *Wyższa Szkoła Handlowa*.

- (2) a. *Biblioteki Głównej Wyższej*  
 library main higher  
 gen:sg:f gen:sg:f gen:sg:f  
*Szkoły Handlowej*  
 school commercial  
 gen:sg:f gen:sg:f
- b. *Biblioteka Główna Wyższej*  
 library main higher  
 nom:sg:f nom:sg:f gen:sg:f  
*Szkoły Handlowej*  
 school commercial  
 gen:sg:f gen:sg:f

While the paper reports detailed figures on named entity recognition performance, the quality of lemmatisation is assessed only for all entity types collectively: “79.6 of the detected NEs were lemmatised correctly” (Piskorski, 2005).

## 3 Phrase lemmatisation as a tagging problem

The idea presented here is directly inspired by Degórski’s observations. First, we will also assume

that lemma of any NP may be obtained by concatenating simple transformations of word forms that make up the phrase. As we will show in Sec. 4, this assumption is virtually always satisfied. We will argue that there is a small finite set of inflectional transformations that are sufficient to lemmatise nearly every Polish NP.

Consider example (1) again. Correct lemmatisation of the phrase may be obtained by applying a series of simple inflectional transformations to each of its words. The first two words need to be turned into nominative forms, the last one is already lemmatised. This is depicted in (3a). To show the real setting, this time we give full NCP tags and word-level lemmas assigned as a result of tagging. In the NCP tagset, the first part of each tag denotes grammatical class (*adj* stands for adjective, *subst* for noun). Adjectives are also specified for degree (*pos* — positive degree).

- (3) a. *głównym*  
*główny*  
*adj:sg:inst:n:pos*  
*miastem*                      *gminy*  
*miasto*                        *gmina*  
*subst:sg:inst:n*    *subst:sg:gen:f*
- b. *główne*                      *miasto*  
*adj:sg:nom:n:pos*    *subst:sg:nom:n*  
*cas=nom*                      *cas=nom*  
*gminy*  
*subst:sg:gen:f*  
 =

Example (3b) consists of three rows: the lemmatised phrase, the desired tags (tags that would be attached to tokens of the lemmatised phrase) and the transformations needed to obtain lemma from orthographic forms. The notation *cas=nom* means that to obtain the desired form (e.g. *główne*) you need to find an entry in a morphological dictionary that bears the same word-level lemma as the inflected form (*główny*) and a tag that results from taking the tag of the inflected form (*adj:sg:inst:n:pos*) and setting the value of the tagset attribute *cas* (grammatical case) to the value *nom* (nominative). The transformation labelled = means that the inflected form is already equal to the desired part of the lemma, hence no transformation is needed.

A tagset note is in order. In the NCP tagset each tag may be decomposed into grammatical class and attribute values, where the choice

of applicable attributes depends on the grammatical class. For instance, nouns are specified for number, gender and case. This assumption is important for our approach to be able to use simple tag transformations in the form *replace the value of attribute A with the new value V (A=V)*. This is not a serious limitation, since the same assumption holds for most tagsets developed for inflectional languages, e.g., the whole MULTEXT-East family (Erjavec, 2012), Czech tagset (Jakubiček et al., 2011).

Our idea is simple: by expressing phrase lemmatisation in terms of word-level transformations we can reduce the task to tagging problem and apply well known Machine Learning techniques that have been devised for solving such problems (e.g. CRF). An important advantage is that this allows to rely not only on the information contained within the phrase to be lemmatised, but also on tokens belonging to its local neighbourhood.

Assuming that we have already trained a statistical model, we need to perform the following steps to obtain lemmatisation of a new text:

1. POS tagging,
2. NP chunking,
3. tagging with transformations by applying the trained model,
4. application of transformations to obtain NP lemmas (using a morphological dictionary to generate forms).

To train the statistical model, we need training data labelled with such transformations. Probably the most reliable way to obtain such data would be to let annotators manually encode a training corpus with such transformations. However, the task would be extremely tedious and the annotators would probably have to undergo special training (to be able to think in terms of transformations). We decided for a simpler solution. The annotators were given a simpler task of assigning each NP instance a lemma and a heuristic procedure was used to induce transformations by matching the manually annotated lemmas to phrases' orthographic forms using a morphological dictionary. The details of this procedure are given in the next section.

We decided to perform the experiments using the data from *Polish Corpus of Wrocław Univer-*

sity of Technology<sup>1</sup> (Broda et al., 2012). The corpus (abbreviated to *KPWr* from now on) contains manual shallow syntactic annotation which includes NP chunks and their syntactic heads. The main motivation to use this corpus was its very permissive licence (Creative Commons Attribution), which will not constrain any further use of the tools developed. What is more, it allowed us to release the data annotated manually with phrase lemmas and under the same licence<sup>2</sup>.

One of the assumptions of *KPWr* annotation is that actual noun phrases and prepositional phrases are labelled collectively as NP chunks. To obtain real noun phrases, phrase-initial prepositions must be stripped off<sup>3</sup>. For practical reasons we decided to include automatic recognition of phrase-initial prepositions into our model: we introduced a special transformation for such cases (labelled *p*), having the interpretation that the token belongs to a phrase-initial preposition and should be discarded when generating phrase lemma. Prepositions are usually contained in single tokens. There are some cases of multi-word units which we treat as prepositions (*secondary prepositions*), e.g. *ze względu na* (*with respect to*). This solution allows to use our lemmatiser directly against chunker output to obtain NP lemmas from both NPs and PPs. For instance, the phrase *o przenoszeniu bakterii drogą płciową* (*about sexual transmission of bacteria*) should be lemmatised to *przenoszenie bakterii drogą płciową* (*sexual transmission of bacteria*).

#### 4 Preparation of training data

First, simple lemmatisation guidelines were developed. The default strategy is to normalise the case to nominative and the number to singular. If the phrase was in fact prepositional, phrase-initial preposition should be removed first. If changing the number would alter semantics of the phrase, it should be left plural (e.g., *warunki* ‘conditions’ as in *terms and conditions*). Some additional exceptions concern pronouns, fixed expressions and

<sup>1</sup>We used version 1.1 downloaded from <http://www.nlp.pwr.wroc.pl/kpwr>.

<sup>2</sup>The whole dataset described in this paper is available at <http://nlp.pwr.wroc.pl/en/static/kpwr-lemma>.

<sup>3</sup>Note that if we decided to use the data from NCP, we would still have to face this issue. Although an explicit distinction is made between NPs and PPs, NPs are not annotated as separate chunks when belonging to a PP chunk (an assumption which is typical for shallow parsing).

proper names. They were introduced to obtain lemmas that are practically most useful.

A subset of documents from *KPWr* corpus was drawn randomly. Each NP/PP belonging to this subset was annotated manually. Contrary to (DeGórski, 2011), we made no exclusions, so the obtained set contains some foreign names and a number of cases which were hard to lemmatise manually. Among the latter there was one group we found particularly interesting. It consisted of items following the following pattern: NP in plural modified by another NP or PP in plural. For many cases it was hard to decide if both parts were to be reverted to singular, only the main one or perhaps both of them should be left in plural. We present two such cases in (4a) and (4b). For instance, (4b) could be lemmatised as *opis tytułu z Wikipedii* (*description of a Wikipedia title*), but it was not obvious if it was better than leaving the whole phrase as is.

- (4) a. *obawy ze strony autorów*  
‘concerns on the part of the authors’  
b. *opisy tytułów z Wikipedii*  
‘descriptions of the Wikipedia titles’

Altogether, the annotated documents contain 1669 phrases. We used the same implementation of the 2+1 model which was used to annotate morphosyntax in NCP (Przepiórkowski and Szwałkiewicz, 2012): two annotators performed the task independently, after which their decisions were compared and the discrepancies were highlighted. The annotators were given a chance to rethink their decisions concerning the highlighted phrases. Both annotators were only told which phrases were lemmatised differently by the other party but they didn’t know the other decision. The purpose of this stage was to correct obvious mistakes. Their output was finally compared, resulting in 94% phrases labelled identically (90% before reconsidering decisions). The remaining discrepancies were decided by a superannotator. The whole set was divided randomly into the development set (1105 NPs) and evaluation set (564 NPs).

The development set was enhanced with word-level transformations that were induced automatically in the following manner. The procedure assumes the usage of a morphological dictionary extracted from Morfeusz SGJP analyser<sup>4</sup> (Woliński,

<sup>4</sup>[morfeusz-sgjp-src-20110416](http://morfeusz-sgjp-src-20110416) package

2006). The dictionary is stored as a set of (*orthographic form, word-level lemma, tag*). The procedure starts with tokenisation of the manually assigned lemma. Next, a heuristic identification of phrase-initial preposition is performed. The assumption is that, having cut the preposition, all the remaining tokens of the original inflected phrase must be matched 1:1 to corresponding tokens from the human-assigned lemma. If any match problem did occur, an error was reported and such a case was examined manually. The only problems encountered were due to proper names unknown to the dictionary and misspelled phrases (altogether about 10 cases). Those cases were dealt with manually. Also, all the extracted phrase-initial prepositions were examined and no controversy was found.

The input and output to the matching procedure is illustrated in Fig. 1. The core matching happens at token level. The task is to find a suitable transformation for the given inflected form from the original phrase, its tag and word-level lemma, but also given the desired form being part of human-assigned lemma. If the inflected form is identical to the desired human-assigned lemma, the '=' transformation is returned without any tag analysis. For other cases the morphological dictionary is required. For instance, the inflected form *tej* tagged as *adj:sg:loc:f:pos* should be matched to the human-assigned form *ta* (the row label *H lem*). The first subtask is to find all entries in the morphological dictionary with the orthographic form equal to human-assigned lemma (*ta*), the word-level lemma equal to the lemma assigned by the tagger (*ten*) and having a tag with the same grammatical class as the tagger has it (*adj*; we deliberately disallow transformations changing the grammatical class). The result is a set of entries with the given lemma and orthographic form, but with different tags attached. For the example considered, two tags may be obtained: *adj:sg:nom:f:pos* and *adj:sg:voc:f:pos* (the former is in nominative case, the latter — in vocative). Each of the obtained tags is compared to the tag attached to the inflected forms (*adj:sg:loc:f:pos*) and this way candidate transformations are generated (*cas=nom* and *cas=voc* here). The transformations are heuristically ranked. Most importantly,

obtained from <http://sgjp.pl/morfeusz/dopobrania.html>. The package is available under 2-clause BSD licence.

*cas=nom* is always preferred, then *nmb=sg* (enforcing singular number), then transforming the gender to different values, preferably to masculine inanimate (*gnd=m3*). The lowest possible ranking is given to a transformation enforcing case value other than nominative.

Original:	<i>przy</i>	<i>tej</i>	<i>drodze</i>
T tags:	<i>prep:</i>	<i>adj:</i>	<i>subst:</i>
	<i>loc</i>	<i>sg:loc:f:pos</i>	<i>sg:loc:f</i>
T lem:	<i>przy</i>	<i>ten</i>	<i>droga</i>
H lem:		<i>ta</i>	<i>droga</i>
Transf.:	<i>p</i>	<i>cas=nom</i>	<i>cas=nom</i>

Figure 1: Matching of an NP and its lemma. The first row shows the original inflected form. The next three present tagger output: tags (split into two rows) and lemmas. *H lem* stands for the lemma assigned by a human. Last row presents the transformations induced.

We are fully aware of limitations of this approach. This ranking was inspired only by intuition obtained from the lemmatisation guidelines and the transformations selected this way may be wrong in a number of cases. While many transformations may lead to obtaining the same lemma for a given form, many of them will still be accidental. Different syncretisms may apply to different lexemes, which can negatively impact the ability of the model to generalise from one phrase to other. On the other hand, manual inspection of some fragments suggest that the transformations inferred are rarely unjustified.

The frequencies of all transformations induced from the development set are given in Tab. 1. Note that the first five most frequent transformation make up 98.7% of all cases. These findings support our hypothesis that a small finite set of transformations is sufficient to express lemmatisation of nearly every Polish NP.

We have also tested an alternative variant of the matching procedure that included additional transformation 'lem' with the meaning *take the word-level lemma assigned by the tagger as the correct lemmatisation*. This transformation could be induced after an unsuccessful attempt to induce the '=' transformation (i.e., if the correct human-assigned lemmatisation was not identical to orthographic form). This resulted in replacing a number of tag-level transformations (mostly *cas=nom*) with the simple 'lem'. The advantage of this vari-

=	2444	72%
cas=nom	434	13%
p	292	9%
nmb=sg	97	3%
cas=nom, nmb=sg	76	2%
gnd=m3	9	
cas=nom, gnd=m3, nmb=sg	7	
gnd=m3, nmb=sg	6	
acn, cas=nom	5	
acm=rec, cas=nom	3	
cas=gen	3	
cas=nom, gnd=m3	3	
cas=nom, gnd=m1	2	
gnd=f, nmb=sg	2	
cas=nom, gnd=f	1	
cas=nom, gnd=f, nmb=sg	1	
cas=nom, nmb=pl	1	
cas=nom, nmb=sg, gnd=m3	1	
<b>Total</b>	<b>3387</b>	<b>100%</b>

Table 1: Frequencies of transformations.

ant is that application of this transformation does not require resorting to the dictionary. The disadvantage is that it is likely to worsen the generalising power of the model.

## 5 CRF and features

The choice of CRF for sequence labelling was mainly influenced by its successful application to chunking of Polish (Radziszewski and Pawlaczek, 2012). The work describes a feature set proposed for this task, which includes word forms in a local window, values of grammatical class, gender, number and case, tests for agreement on number, gender and case, as well as simple tests for letter case.

We took this feature set as a starting point. Then we performed some experiments with feature generation and selection. For this purpose the development set was split into training and testing part. The most obvious, yet most successful change was to introduce features returning the chunk tag assigned to a token. As KPWr also contains information on the location of chunks' syntactic heads and this information is also output by the chunker, we could also use this in our features. Another improvement resulted from completely removing tests for grammatical gender and limiting the employed tests for number to the current token.

The final feature set includes the following

items:

- the word forms (turned lower-case) of tokens occupying a local window  $(-2, \dots, +2)$ ,
- word form bigrams:  $(-1, 0)$  and  $(0, 1)$ ,
- chunk tags (IOB2 tags concatenated with Boolean value denoting whether the syntactic head is placed at the position), for a local window  $(-1, 0, +1)$
- chunk tags (IOB2 tags only) for positions  $-2$  and  $+2$ , and two chunk tag bigrams:  $(-1, 0)$  and  $(0, 1)$ ,
- grammatical class of tokens in the window  $(-2, \dots, +2)$ ,
- grammatical class for the focus token (0) concatenated with the last character of the word-form,
- values of grammatical case for tokens  $(-2, -1, +1, +2)$ ,
- grammatical class of the focus token concatenated with its gender value,
- 2-letter prefix of the word form (lower-cased),
- tests for agreements and letter case as in (Radziszewski and Pawlaczek, 2012).

## 6 Evaluation

The performed evaluation assumed training of the CRF on the whole development set annotated with the induced transformations and then applying the trained model to tag the evaluation part with transformations. Transformations were then applied and the obtained phrase lemmas were compared to the reference annotation. This procedure includes the influence of deficiencies of the morphological dictionary. The version of KPWr used here was tagged automatically using the WCRFT tagger (Radziszewski, 2013), hence tagging errors are also included.

Degórski (2011) reports separate figures for the performance of the entire system (chunker + NP lemmatiser) on the whole test set and performance of the entire system limiting the test set only to those phrases that the system is able to chunk correctly (i.e., to output correct phrase boundaries). Such a choice is reasonable given that his system

is based on rules that intermingle chunking with lemmatisation. We cannot expect the system to lemmatise correctly those groups which it is unable to capture. Our approach assumes two-stage operation, where the chunker stage is partially independent from the lemmatisation. This is why we decided to report performance of the whole system on the whole test set, but also, performance of the lemmatisation module alone on the *whole test set*. This seems more appropriate, since the chunker may be improved or completely replaced independently, while discarding the phrases that are too hard to parse is likely to bias the evaluation of the lemmatisation stage (what is hard to chunk is probably also hard to lemmatise).

For the setting where chunker was used, we used the CRF-based chunker mentioned in the previous section (Radziszewski and Pawlaczek, 2012). The chunker has been trained on the entire KPWr except for the documents that belong to the evaluation set.

Degórski (2011) uses concatenation of word-level base forms assigned by the tagger as a baseline. Observation of the development set suggests that returning the original inflected NPs may be a better baseline. We tested both variants. As detection of phrase-initial prepositions is a part of our task formulation, we had to implement it in the baseline algorithms as well. Otherwise, the comparison would be unfair. We decided to implement both baseline algorithms using the same CRF model but trained on fabricated data. The training data for the ‘take-orthographic-form’ baseline was obtained by leaving the ‘remove-phrase-initial-preposition’ (‘p’) transformation and replacing all others with ‘=’. Similarly, for the ‘take-lemma’ baseline, other transformations were substituted with ‘lem’.

The results of the full evaluation are presented in Tab. 2. The first conclusion is that the figures are disappointingly low, but comparable with the 58.5% success rate reported in (Degórski, 2011). The other observation is that the proposed solution significantly outperforms both baseline, out of which the ‘take-orthographic-form’ (*orth baseline*) performs slightly better. Also, it turns out that the variation of the matching procedure using the ‘lem’ transformation (row labelled *CRF lem*) performs slightly worse than the procedure without this transformation (row *CRF nolem*). This supports the suspicion that relying on word-

level lemmas may reduce the ability to generalise.

Algorithm	Prec.	Recall	F
CRF nolem	55.1%	56.9%	56.0%
CRF lem	53.7%	55.5%	54.6%
orth baseline	38.6%	39.9%	39.2%
lem baseline	36.2%	37.4%	36.8%

Table 2: Performance of NP lemmatisation including chunking errors.

Results corresponding to performance of the lemmatisation module alone are reported in Tab. 3. The test has been performed using chunk boundaries and locations of syntactic heads taken from the reference corpus. In this settings recall and precision have the same interpretation, hence we simply refer to the value as *accuracy* (percentage of chunks that were lemmatised correctly). The figures are considerably higher than those reported in Tab. 2, which shows the huge impact of chunking errors. It is worth noting that the best accuracy achieved is only slightly lower than that achieved by Degórski (82.9%), while our task is harder. As mentioned above, in Degórski’s setting, the phrases that are too hard to parse are excluded from the test set. Those phrases are also likely to be hard cases for lemmatisation. The other important difference stems from phrase definitions used in both corpora; NPs in NCP are generally shorter than the chunks allowed in KPWr. Most notably, KPWr allows the inclusion of PP modifiers within NP chunks (Broda et al., 2012). It seems likely that the proposed algorithm would performed better when trained on data from NCP which assumes simpler NP definition. Note that the complex NP definition in KPWr also explains the huge gap between results of lemmatisation alone and lemmatisation including chunking errors.

Algorithm	Correct lemmas	Accuracy
CRF nolem	455 / 564	80.7%
CRF lem	444 / 564	78.7%
orth baseline	314 / 564	55.7%
lem baseline	290 / 564	51.4%

Table 3: Performance of NP lemmatisation alone.

We also checked the extent to which the entries unknown to the morphological dictionary could lower the performance of lemmatisation. It turned out that only 8 words couldn’t be transformed during evaluation due to lack of the entries that

were sought in the morphological dictionary, out of which 5 were anyway handled correctly in the end by using the simple heuristic to output the ‘=’ transformation when everything else fails.

A rudimentary analysis of lemmatiser output indicates that the most common error is the assignment of the orthographic form as phrase lemma where something else was expected. This seems to concern mostly many NPs that are left in plural, even simple ones (e.g. *audycje telewizyjne* ‘TV programmes’), but there are also some cases of personal pronouns left in oblique case (*was* ‘you-pl-accusative/genitive’). It seems that a part of these cases come from tagging errors (even if the correct transformation is obtained, the results of its application depend on the tag and lemma attached to the inflected form by the tagger). Not surprisingly, proper names are hard cases for the model (e.g. *Pod Napięciem* was lemmatised to *napięcie*, which would be correct weren’t it a title).

## 7 Conclusions and further work

We presented a novel approach to lemmatisation of Polish noun phrases. The main advantage of this solution is that it allows to separate the lemmatisation phrase from the chunking phrase. Degórski’s rule-based approach (Degórski, 2011) was also built on top of an existing parser but, as he notes, to improve the lemmatisation accuracy, the grammar underlying the parser should actually be rewritten with lemmatisation in mind. The other advantage of the approach presented here is that it is able to learn from a corpus containing manually assigned phrase lemmas. Extending existing chunk-annotated corpora with phrase lemmas corresponds to a relatively simple annotation task.

The performance figures obtained by our algorithm are comparable with that of Degórski’s grammar, while the conditions under which our system was evaluated were arguably less favourable. To enable a better comparison it would be desirable to evaluate our approach against the phrases from NCP.

The main disadvantage of the approach lies in the data preparation stage. It requires some semi-manual work to obtain labelling with transformations, which is language- and tagset-dependent. A very interesting alternative has been suggested by an anonymous reviewer: instead of considering tag-level transformations that require an exhaustive morphological dictionary, it would be simpler

to rely entirely on string-to-string transformations that map inflected forms to their expected counterparts. Such transformations may be expressed in terms of simple edit scripts, which has already been successfully applied to word-level lemmatisation of Polish and other languages (Chrupała et al., 2008). This way, the training data labelled with transformations could be obtained automatically. What is more, application of such transformations also does not depend on the dictionary. It is not obvious how this would affect the performance of the module and, hence, needs to be evaluated. We plan this as our further work.

Also, it would be worthwhile to evaluate the presented solution for other Slavic languages.

## Acknowledgments

This work was financed by Innovative Economy Programme project POIG.01.01.02-14-013/09.

## References

- Robert Bembenik, Łukasz Skonieczny, Henryk Rybiński, Marzena Kryszkiewicz, and Marek Niezgódka, editors. 2013. *Intelligent Tools for Building a Scientific Information Platform*, volume 467 of *Studies in Computational Intelligence*. Springer Berlin Heidelberg.
- Bartosz Broda, Michał Marcińczuk, Marek Maziarz, Adam Radziszewski, and Adam Wardyński. 2012. KPWr: Towards a free corpus of Polish. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of LREC’12*, Istanbul, Turkey. ELRA.
- Aleksander Buczyński and Adam Przepiórkowski. 2009. Human language technology. challenges of the information society. chapter Spejd: A Shallow Processing and Morphological Disambiguation Tool, pages 131–141. Springer-Verlag, Berlin, Heidelberg.
- Grzegorz Chrupała, Georgiana Dinu, and Josef van Genabith. 2008. Learning morphology with Morfette. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, and Daniel Tapias, editors, *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC’08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA).
- Łukasz Degórski. 2011. Towards the lemmatisation of Polish nominal syntactic groups using a shallow

- grammar. In Pascal Bouvry, Mieczysław A. Kłopotek, Franck Lèprevost, Małgorzata Marciniak, Agnieszka Mykowiecka, and Henryk Rybiński, editors, *Security and Intelligent Information Systems: International Joint Conference, SIIS 2011, Warsaw, Poland, June 13-14, 2011, Revised Selected Papers*, volume 7053 of *Lecture Notes in Computer Science*, pages 370–378. Springer-Verlag.
- Tomaž Erjavec. 2012. MULTEXT-East: morphosyntactic resources for Central and Eastern European languages. *Language Resources and Evaluation*, 46(1):131–142.
- Miloš Jakubiček, Vojtěch Kovář, and Pavel Šmerk. 2011. Czech morphological tagset revisited. In *Proceedings of Recent Advances in Slavonic Natural Language Processing*, pages 29–42, Brno.
- Jan Kocoń and Maciej Piasecki. 2012. Heterogeneous named entity similarity function. In Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala, editors, *Text, Speech and Dialogue*, volume 7499 of *Lecture Notes in Computer Science*, pages 223–231. Springer Berlin Heidelberg.
- Małgorzata Marciniak and Agnieszka Mykowiecka. 2013. Terminology extraction from domain texts in Polish. In Bembenik et al. (Bembenik et al., 2013), pages 171–185.
- Jakub Piskorski, Karol Wieloch, and Marcin Sydow. 2009. On knowledge-poor methods for person name matching and lemmatization for highly inflectional languages. *Information Retrieval*, 12(3):275–299.
- Jakub Piskorski. 2005. Named-entity recognition for Polish with SProUT. In Leonard Bolc, Zbigniew Michalewicz, and Toyooki Nishida, editors, *Intelligent Media Technology for Communicative Intelligence*, volume 3490 of *Lecture Notes in Computer Science*, pages 122–133. Springer Berlin Heidelberg.
- Adam Przepiórkowski. 2007. Slavic information extraction and partial parsing. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing*, pages 1–10, Praga, Czechy, June. Association for Computational Linguistics.
- Adam Przepiórkowski. 2009. A comparison of two morphosyntactic tagsets of Polish. In Violetta Koseska-Toszewa, Ludmila Dimitrova, and Roman Roszko, editors, *Representing Semantics in Digital Lexicography: Proceedings of MONDILEX Fourth Open Workshop*, pages 138–144, Warszawa.
- Adam Przepiórkowski and Łukasz Szalkiewicz. 2012. Anotacja morfoskładniowa. In Adam Przepiórkowski, Mirosław Bańko, Rafał L. Górski, and Barbara Lewandowska-Tomaszczyk, editors, *Narodowy Korpus Języka Polskiego*. Wydawnictwo Naukowe PWN, Warsaw.
- Adam Radziszewski and Adam Pawlaczek. 2012. Large-scale experiments with NP chunking of Polish. In *Proceedings of the 15th International Conference on Text, Speech and Dialogue*, Brno, Czech Republic. Springer Verlag.
- Adam Radziszewski. 2013. A tiered CRF tagger for Polish. In Bembenik et al. (Bembenik et al., 2013), pages 215–230.
- Peter Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2:303–336.
- Marcin Woliński. 2006. Morfeusz — a practical tool for the morphological analysis of Polish. In Mieczysław A. Kłopotek, Sławomir T. Wierzchoń, and Krzysztof Trojanowski, editors, *Proceedings of IIPWM'06*, pages 511–520, Ustroń, Poland, June 19–22. Springer-Verlag, Berlin.

# Using Conceptual Class Attributes to Characterize Social Media Users

Shane Bergsma and Benjamin Van Durme

Department of Computer Science and Human Language Technology Center of Excellence  
Johns Hopkins University  
Baltimore, MD 21218, USA

## Abstract

We describe a novel approach for automatically predicting the hidden demographic properties of social media users. Building on prior work in common-sense knowledge acquisition from third-person text, we first learn the distinguishing attributes of certain classes of people. For example, we learn that people in the *Female* class tend to have *maiden names* and *engagement rings*. We then show that this knowledge can be used in the analysis of first-person communication; knowledge of distinguishing attributes allows us to both classify users and to bootstrap new training examples. Our novel approach enables substantial improvements on the widely-studied task of user gender prediction, obtaining a 20% relative error reduction over the current state-of-the-art.

## 1 Introduction

There has been growing interest in *characterizing* social media users based on the content they generate; that is, automatically labeling users with demographic categories such as age and gender (Burger and Henderson, 2006; Schler et al., 2006; Rao et al., 2010; Mukherjee and Liu, 2010; Penacchiotti and Popescu, 2011; Burger et al., 2011; Van Durme, 2012). Automatic user characterization has applications in targeted advertising and personalization, and could also lead to finer-grained assessment of public opinion (O'Connor et al., 2010) and health (Paul and Dredze, 2011).

Consider the following tweet and suppose we wish to predict the user's gender:

*Dirac was one of my boyhood heroes. I'm glad I met him once. RT Paul Dirac image by artist Eric Handy: [http:...](#)*

State-of-the-art approaches cast this problem as a classification task and train classifiers using supervised learning (Section 2). The features of the classifier are indicators of specific words in the user-generated text. While a human would assume that someone with *boyhood heroes* is **male**, a standard classifier has no way of exploiting such knowledge unless the phrase occurs in training data. We present an algorithm that improves user characterization by collecting and exploiting such common-sense knowledge.

Our work is inspired by algorithms that process large text corpora in order to discover the attributes of semantic classes, e.g. (Berland and Charniak, 1999; Schubert, 2002; Almuhareb and Poesio, 2004; Tokunaga et al., 2005; Girju et al., 2006; Paşca and Van Durme, 2008; Alfonseca et al., 2010). We learn the distinguishing attributes of different demographic groups (Section 3), and then automatically assign users to these groups whenever they refer to a distinguishing attribute in their writings (Section 4). Our approach obviates the need for expensive annotation efforts, and allows us to rapidly bootstrap training data for new classification tasks.

We validate our approach by advancing the state-of-the-art on the most well-studied user classification task: predicting user gender (Section 5). Our bootstrapped system, trained purely from automatically-annotated Twitter data, significantly reduces error over a state-of-the-art system trained on thousands of gold-standard training examples.

## 2 Supervised User Characterization

The current state-of-the-art in user characterization is to use supervised classifiers trained on annotated data. For each instance to be classified, the output is a decision about a distinct demographic property, such as *Male/Female* or *Over/Under-18*. A variety of classification algorithms have been employed, including SVMs (Rao et al., 2010), de-

cision trees (Pennacchiotti and Popescu, 2011), logistic regression (Van Durme, 2012), and the Winnow algorithm (Burger et al., 2011).

**Content Features: *BoW*** Prior classifiers use a set of features encoding the presence of specific words in the user-generated text. We call these features *BoW* features as they encode the standard Bag-of-Words representation which has been highly effective in text categorization and information retrieval (Sebastiani, 2002).

**User-Profile Features: *U<sub>sr</sub>*** Some researchers have explored features for user-profile meta-information in addition to user content. This may include the user’s communication behavior and network of contacts (Rao et al., 2010), their full name (Burger et al., 2011) and whether they provide a profile picture (Pennacchiotti and Popescu, 2011). We focus on the case where we only have access to the user’s screen-name (a.k.a. username). Using a combination of content and username features “represents a use case common to many different social media sites, such as chat rooms and news article comment streams” (Burger et al., 2011). We refer to features derived from a username as *U<sub>sr</sub>* features in our experiments.

### 3 Learning Class Attributes

We aim to improve the automated classification of users into various demographic categories by learning and applying the distinguishing attributes of those categories, e.g. that *males* have *boyhood heroes*. Our approach builds on lexical-semantic research on the topic of *class-attribute extraction*. In this research, the objective is to discover various *attributes* or *parts* of classes of entities. For example, Berland and Charniak (1999) learn that the class *car* has parts such as *headlight*, *windshield*, *dashboard*, etc. Berland and Charniak extract these attributes by mining a corpus for fillers of patterns such as ‘*car*’s *X*’ or ‘*X* of a *car*’. Note their patterns explicitly include the class itself (*car*). Another approach is to use patterns that are based on *instances* (i.e. hyponyms or sub-classes) of the class. For example, Paşca and Van Durme (2007) learn the attributes of the class *car* via patterns involving instances of cars, e.g. *Chevrolet Corvette*’s *X* and *X* of a *Honda Civic*. For these approaches, lists of instances are typically collected from publicly-available resources such as WordNet or Wikipedia (Paşca and Van Durme, 2007;

Van Durme et al., 2008), acquired automatically from corpora (Paşca and Van Durme, 2008; Alfonso et al., 2010), or simply specified by hand (Schubert, 2002).

**Creation of Instance Lists** We use an instance-based approach; our instances are derived from collections of common nouns that are associated with roles and occupations of people. For the gender task that we study in our experiments, we acquire class instances by filtering the dataset of nouns and their genders created by Bergsma and Lin (2006). This dataset indicates how often a noun is referenced by a male, female, neutral or plural pronoun. We extract prevalent common nouns for males and females by selecting only those nouns that (a) occur more than 200 times in the dataset, (b) mostly occur with male or female pronouns, and (c) occur as lower-case more often than upper-case in a web-scale N-gram corpus (Lin et al., 2010). We then classify a noun as **Male** (resp. **Female**) if the noun is indicated to occur with male (resp. female) pronouns at least 85% of the time. Since the gender data is noisy, we also quickly pruned by hand any instances that were malformed or obviously incorrectly assigned by our automatic process. This results in 652 instances in total. Table 1 provides some examples.

**Male:** bouncer, altar boy, army officer, dictator, assailant, cameraman, drifter, chauffeur, bad guy

**Female:** young lady, lesbian, ballerina, waitress, granny, chairwoman, heiress, soprano, socialite

Table 1: Example instances used for extraction of class attributes for the gender classification task

**Attribute Extraction** We next collect and rank attributes for each class. We first look for fillers of attribute-patterns involving each of the instances. Let *I* represent an instance of one of our classes. We find fillers of the single high-precision pattern:

$$\underbrace{\{\text{word}=I, \text{tag}=NN\}}_{\text{instance}} \underbrace{\{\text{word}='s\}}_{\text{'s}} \underbrace{\{\{\text{word}=. * \}^* \{\text{tag}=N. * \}\}}_{\text{attribute}}$$

(E.g. *dictator*’s [*former mistress*]). The expression “tag=NN” means that *I* must be tagged as a noun. The expression in square brackets is the filler, i.e. the extracted attribute, *A*. The notation “{word=. \*} \* tag=N. \*” means that *A* can be any sequence of tokens ending in a noun. We use an

equivalent pattern when  $I$  is multi-token. The output of this process is a set of  $(I, A)$  pairs.

In attribute extraction, typically one must choose between the precise results of rich patterns (involving punctuation and parts-of-speech) applied to small corpora (Berland and Charniak, 1999) and the high-coverage results of superficial patterns applied to web-scale data, e.g. via the Google API (Almuhareb and Poesio, 2004). We obtain the best of both worlds by matching our precise pattern against a version of the Google N-gram Corpus that includes the part-of-speech tag distributions for every N-gram (Lin et al., 2010). We found that applying this pattern to web-scale data is effective in extracting useful attributes. We acquired around 20,000 attributes in total.

**Finding Distinguishing Attributes** Unlike prior work, we aim to find *distinguishing* properties of each class; that is, the kinds of properties that uniquely distinguish a particular category. Prior work has mostly focused on finding “relevant” attributes (Alfonseca et al., 2010) or “correct” parts (Berland and Charniak, 1999). A *leg* is a relevant and correct part of both a male and a female (and many other living and inanimate objects), but it does not help us distinguish males from females in social media. We therefore rank our attributes for each class by their strength of association with instances of that specific class.<sup>1</sup>

To calculate the association, we first disregard the count of each  $(I, A)$  pair and consider each unique pair to be a single probabilistic event. We then convert the  $(I, A)$  pairs to corresponding  $(C, A)$  pairs by replacing  $I$  with the corresponding class,  $C$ . We then calculate the pointwise mutual information (Church and Hanks, 1990) between each  $C$  and  $A$  over the set of events:

$$\text{PMI}(C, A) = \log \frac{p(C, A)}{p(C)p(A)} \quad (1)$$

If the  $\text{PMI} > 0$ , the observed probability of a class and attribute co-occurring is greater than the probability of co-occurrence that we would expect if  $C$  and  $A$  were independently distributed. For each class, we rank the attributes by their PMI scores.

<sup>1</sup>Reisinger and Paşca (2009) considered the related problem of finding the most appropriate *class* for each *attribute*; they take an existing ontology of concepts (WordNet) as a class hierarchy and use a Bayesian approach to decide “the correct level of abstraction for each attribute.”

**Filtering Attributes** We experimented with two different methods to select a final set of distinguishing attributes for each class: (1) we used a threshold to select the top-ranked attributes for each class, and (2) we manually filtered the attributes. For the gender classification task, we manually filtered the entire set of attributes to select around 1000 attributes that were judged to be discriminative (two thirds of which are female). This filtering took one annotator only a few hours to complete. Because this process was so trivial, we did not invest in developing annotation guidelines or measuring inter-annotator agreement. We make these filter attributes available online as an attachment to this article, available through the ACL Anthology.

Ultimately, we discovered that manual filtering was necessary to avoid certain pathological cases in our Twitter data. For example, our PMI scoring finds *homepage* to be strongly associated with males. In our gold-standard gender data (Section 5), however, *every* user has a homepage [by dataset construction]; we might therefore incorrectly classify every user as **Male**. We agree with Richardson et al. (1998) that “automatic procedures ... provide the only credible prospect for acquiring world knowledge on the scale needed to support common-sense reasoning” but “hand vetting” might be needed to ensure “accuracy and consistency in production level systems.” Since our approach requires manual involvement in the filtering of the attribute list, one might argue that one should simply manually enumerate the most relevant attributes directly. However, the manual generation of conceptual features by a single researcher results in substantial variability both across and within participants (McRae et al., 2005). Psychologists therefore generate such lists by pooling the responses across many participants: future work may compare our “automatically generate, manually prune” approach to soliciting attributes via crowdsourcing.<sup>2</sup>

Table 2 gives examples of our extracted at-

<sup>2</sup>One can also view the work of manually filtering attributes as a kind of “feature labeling.” There is evidence from Zaidan et al. (2007) that a few hours of feature labeling can be more productive than annotating new training examples. In fact, since Zaidan et al. (2007) label features at the token level (e.g., in our case one would highlight “handbag” in a given tweet), while we label features at the type level (e.g., deciding whether to mark the word “handbag” as feminine in general), our process is likely even more efficient. Future work may also wish to consider this connection to so-called “annotator rationales” more deeply.

**Male:** wife, widow, wives, ex-girlfriend, erection, testicles, wet dream, bride, buddies, ex-wife, first-wife, penis, death sentence, manhood

**Female:** vagina, womb, maiden name, dresses, clitoris, wedding dress, uterus, shawl, necklace, ex-husband, ex-boyfriend, dowry, nightgown

Table 2: Example attributes for gender classes, in descending order of class-association score

tributes. Our approach captures many *multi-token* attributes; these are often distinguishing even though the head noun is ambiguous (e.g. *name* is ambiguous, *maiden name* is not). Our attributes also go beyond the traditional meronyms that were the target of earlier work. As we discuss further in Related Work (Section 7), previous researchers have worried about a proper definition of *parts* or *attributes* and relied on human judgments for evaluation (Berland and Charniak, 1999; Girju et al., 2006; Van Durme et al., 2008). For us, whether a property such as *dowry* should be considered an “attribute” of the class **Female** is immaterial; we echo Almuhareb and Poesio (2004) who (on a different task) noted that “while the notion of ‘attribute’ is not completely clear... our results suggest that trying to identify attributes is beneficial.”

#### 4 Applying Class Attributes

To classify users using the extracted attributes, we look for cases where users refer to such attributes in their first-person writings. We performed a preliminary analysis of a two-week sample of tweets from the TREC Tweets2011 Corpus.<sup>3</sup> We found that users most often reveal their attributes in the possessive construction, “my *X*” where *X* is an attribute, quality or event that they possess (in a linguistic sense). For example, we found over 1000 tweets with the phrase “my wife.” In contrast, “I have a wife” occurs only 5 times.<sup>4</sup>

We therefore assign a user to a demographic category as follows: We first part-of-speech tag our data using CRFTagger (Phan, 2006) and then look for “my *X*” patterns where *X* is a sequence of tokens terminating in a noun, analogous to our

<sup>3</sup><http://trec.nist.gov/data/tweets/> This corpus was developed for the TREC Microblog track (Soboroff et al., 2012).

<sup>4</sup>Note that “I am a man” occurs only 20 times. Users also reveal their names in “my name is *X*” patterns in several hundred tweets, but this is small compared to cases of self-distinguishing *attributes*. Exploiting these alternative patterns could nevertheless be a possible future direction.

attribute-extraction pattern (Section 3).<sup>5</sup> When a user uses such a “my *X*” construction, we match the filler *X* against our attribute lists for each class. If the filler is on a list, we call it a *self-distinguishing attribute* of a user. We then apply our knowledge of the self-distinguishing attribute and its corresponding class in one of the following three ways:

##### (1) *ARules*: Using Attribute-Based Rules to Override a Classifier

When human-annotated data is available for training and testing a supervised classifier, we refer to it as *gold standard* data. Our first technique provides a simple way to use our identified self-distinguishing attributes in conjunction with a classifier trained on gold-standard data. If the user has any self-distinguishing attributes, we assign the user to the corresponding class; otherwise, we trust the output of the classifier.

##### (2) *Bootstrapped*: Automatic Labeling of Training Examples

Even without gold standard training data, we can use our self-distinguishing attributes to automatically bootstrap annotations. We collect a large pool of unlabeled users and their tweets, and we apply the *ARules* described above to label those users that have self-distinguishing attributes. Once an example is auto-annotated, we delete the self-distinguishing attributes from the user’s content. This prevents the subsequent learning algorithm from trivially learning the rules with which we auto-annotated the data. Next, the auto-annotated examples are used as training data for a supervised system.<sup>6</sup> Finally, when applying the *Bootstrapped* classifiers, we can still apply the *ARules* as a post-process (although in practice this made little difference in our final results).

##### (3) *BootStacked*: Gold Standard and *Bootstrapped* Combination

Although we show that an accurate classifier can be trained using auto-annotated *Bootstrapped* data alone, we also test whether we can combine this data with any gold-standard training examples to achieve even better performance. We use the following simple but

<sup>5</sup>While we used an “off the shelf” POS tagger in this work, we note that taggers optimized specifically for social media are now available and would likely have resulted in higher tagging accuracy (e.g. Owoputi et al. (2013)).

<sup>6</sup>Note that while our target gender task presents mutually-exclusive output classes, we can still train classifiers for other categories without clear opposites (e.g. for labeling users as *Parents* or *Doctors*) by using the 1-class classification paradigm (Koppel and Schler, 2004).

effective method for combining data from these two sources, inspired by prior techniques used in the domain adaptation literature (Daumé III and Marcu, 2006). We first use the trained *Bootstrapped* system to make predictions on the entire set of gold standard data (gold train, development, and test sets). We then use these predictions as *features* in a classifier trained on the gold standard data. We refer to this system as the *BootStacked* system in our evaluation.

## 5 Twitter Gender Prediction

To test the use of self-distinguishing attributes in user classification, we apply our methods to the task of gender classification on Twitter. This is an important and intensely-studied task within academia and industry. Furthermore, for this task it is possible to semi-automatically acquire large amounts of ground truth (Burger et al., 2011). We can therefore benchmark our approach against state-of-the-art supervised systems trained with plentiful gold-standard data, giving us an idea of how well our *Bootstrapped* system might compare to theoretically top-performing systems on other tasks, domains, and social media platforms where such gold-standard training data is not available.

**Gold Data** Our data is derived from the corpus created by Burger et al. (2011). Burger et al. observed that many Twitter users link their Twitter profile to homepages on popular blogging websites. Since “many of these [sites] have well-structured profile pages [where users] must select gender and other attributes from dropdown menus,” they were able to link these attributes to the Twitter users. Using this process, they created a large multi-lingual corpus of Twitter users and genders.

We filter non-English tweets from this corpus using the LID system of Bergsma et al. (2012) and also tweets containing URLs (since many of these are spam) and re-tweets. We then filter users with <40 tweets and randomly divide the remaining users into 2282 training, 1140 development, and 1141 test examples.

**Classifier Set-up** We train logistic-regression classifiers on this gold standard data via the LIBLINEAR package (Fan et al., 2008). We optimize the classifier’s regularization parameter on development data and report final results on the held-out test examples. We also report the results of

our new attribute-based strategies (Section 4) on the test data. We report *accuracy*: the percentage of examples labeled correctly.

Our classifiers use both *BoW* and *Usr* features (Section 2). To increase the generality of our *BoW* features, we preprocess the text by lower-casing and converting all digits to special ‘#’ symbols. We then create real-valued features that encode the *log*-count of each word in the input. While Burger et al. (2011) found “no appreciable difference in performance” when using either binary presence/absence features or encoding the frequency of the word, we found real-valued features worked better in development experiments. For the *Usr* features, we add special beginning and ending characters to the username, and then create features for all character n-grams of length two-to-four in the modified username string. We include n-gram features with the original capitalization pattern and separate features with the n-grams lower-cased.

**Unlabeled Data** For *Bootstrapped* training, we also use a pool of unlabeled Twitter data. This pool comprises the union of 2.2 billion tweets from 05/2009 to 10/2010 (O’Connor et al., 2010), 1.9 billion tweets collected from 07/2011 to 11/2012, and 80 million tweets collected from the followers of 10-thousand location and language-specific Twitter feeds. We filter this corpus as above, except we do not put any restrictions on the number of tweets needed per user. We also filter any users that overlap with our gold standard data.

**Bootstrapping Analysis** We apply our *Bootstrapped* auto-annotation strategy to this unlabeled data, yielding 789,285 auto-annotated examples of users and their tweets. The decisions of our bootstrapping process reflect the true gender distribution; the auto-annotated data is 60.5% **Female**, remarkably close to the 60.9% proportion in our gold standard test set. Figure 1 shows that a wide range of self-distinguishing attributes are used in the auto-annotation process. This is important because if only a few attributes are used (e.g. *wife/husband* or *penis/vagina*), we might systematically miss a segment of users (e.g. young people that don’t have husbands or wives, or people that don’t frequently talk about their genitalia). Thus a wide range of common-sense knowledge is useful for bootstrapping, which is one reason why automatic approaches are needed to acquire it.

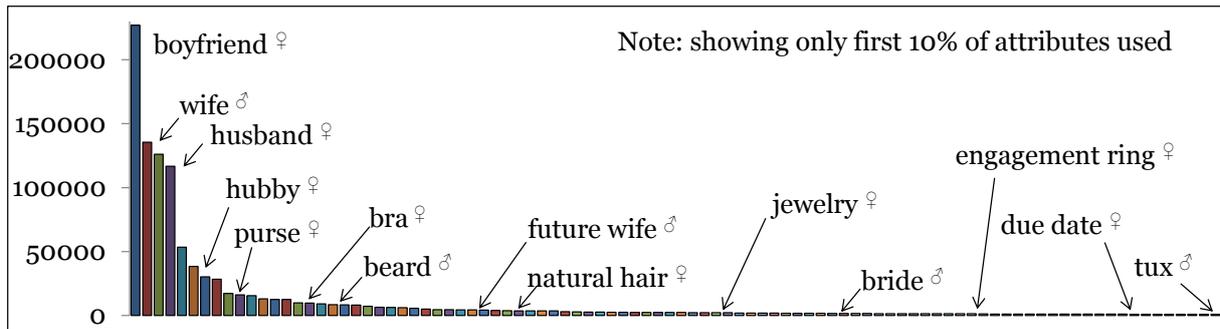


Figure 1: Frequency with which attributes are used to auto-annotate examples in the bootstrapping approach. The plot identifies some attributes and their corresponding class (labeled via gender symbol).

Majority-class baseline	60.9
Supervised on 100 examples	72.0
Supervised on 2282 examples	84.0
Supervised on 100 examples + <i>ARules</i>	74.7
Supervised on 2282 examples + <i>ARules</i>	84.7
<i>Bootstrapped</i>	86.0
<i>BootStacked</i>	<b>87.2</b>

Table 3: Classification accuracy (%) on gold standard test data for user gender prediction on Twitter

## 6 Results

Our main classification results are presented in Table 3. The majority-class baseline for this task is to always choose **Female**; this achieves an accuracy of 60.9%. A standard classifier trained on 100 gold-standard training examples improves over this baseline, to 72.0%, while one with 2282 training examples achieves 84.0%. This latter result represents the current state-of-the-art: a classifier trained on thousands of gold standard examples, making use of both *Usr* and *BoW* features. Our performance compares favourably to Burger et al. (2011), who achieved 81.4% using the same features, but on a very different subset of the data (also including tweets in other languages).<sup>7</sup>

Applying the *ARules* as a post-process significantly improves performance in both cases (McNemar’s,  $p < 0.05$ ). It is also possible to use the *ARules* as a stand-alone system rather than as a post-process, however the coverage is low: we find a distinguishing attribute in 18.3% of the 695 **Female** instances in the test data, and make the cor-

<sup>7</sup>Note that it is possible to achieve even higher performance on gender classification in social media if you have further information about a user, such as their full first and last name (Burger et al., 2011; Bergsma et al., 2013).

rect decision in 96.9% of these cases. We find a distinguishing attribute in 11.4% of the 446 **Male** instances, with 86.3% correct decisions.

The *Bootstrapped* system substantially improves over the state-of-the-art, achieving 86% accuracy and doing so *without using any gold standard training data*. This is important because having thousands of gold standard annotations for every possible user characterization task, in every domain and social media platform, is not realistic. Combining the bootstrapped classifier with the gold standard annotations in the *BootStacked* model results in further gains in performance.<sup>8</sup> These results provide strong validation for both the inherent utility of class-attributes knowledge in user characterization and the effectiveness of our specific strategies for exploiting such knowledge.

Figure 2 shows the learning curve of the *Bootstrapped* classifier. Performance rises consistently across all the auto-annotated training data; this is encouraging because there is theoretically no reason not to vastly increase the amount of auto-annotated data by collecting an even larger collection of tweets. Finally, note that most of the gains of the *Bootstrapped* system appear to derive from the tweet content itself, i.e. the *BoW* features. However, the *Usr* features are also helpful at most training sizes.

We provide some of the top-ranked features of the *Bootstrapped* system in Table 4. We see that a variety of other common-sense knowledge is learned by the system (e.g., the association between males and urinals, boxers, fatherhood, etc.), as well as stylistic clues (e.g. **Female** users using *betcha* and *xox* in their writing). The username

<sup>8</sup>We observed no further gains in accuracy when applying the *ARules* as a post-process on top of these systems.

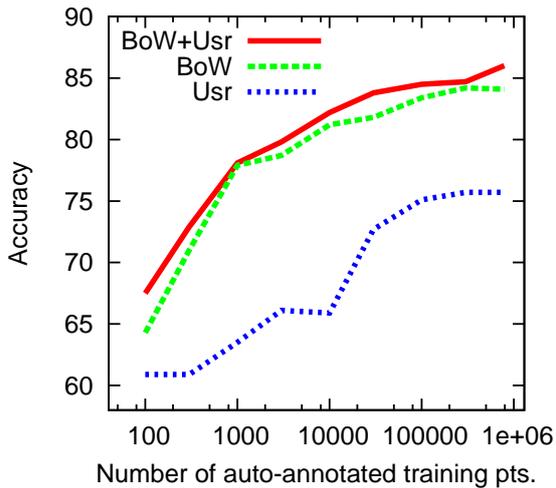


Figure 2: Learning curve for *Bootstrapped* logistic-regression classifier, with automatically-labeled data, for different feature classes.

features capture reasonable associations between gender classes and particular names (such as *mike*, *tony*, *omar*, etc.) and also between gender classes and common nouns (such as *guy*, *dad*, *sir*, etc.).

## 7 Related Work

**User Characterization** The field of sociolinguistics has long been concerned with how various morphological, phonological and stylistic aspects of language can vary with a person’s age, gender, social class, etc. (Fischer, 1968; Labov, 1972). This early work therefore had an emphasis on analyzing the *form* of language, as opposed to its *content*. This emphasis continued into early machine learning approaches, which predicted author properties based on the usage of function words, parts-of-speech, punctuation (Koppel et al., 2002) and spelling/grammatical errors (Koppel et al., 2005).

Recently, researchers have focused less on the sociolinguistic implications and more on the tasks themselves, naturally leading to classifiers with feature representations capturing content in addition to style (Schler et al., 2006; Garera and Yarowsky, 2009; Mukherjee and Liu, 2010). Our work represents a logical next step for content-based classification, a step partly suggested by Schler et al. (2006) who noted that “those who are interested in automatically profiling bloggers for commercial purposes would be well served by considering additional features - which we deliberately ignore in this study - such as author self-identification.”

**Male *BoW* features:** wife, wifey, sucked, shave, boner, boxers, missus, installed, manly, in-laws, brah, urinal, kickoff, golf, comics, ubuntu, homo, nhl, jedi, fatherhood, nigga, movember, algebra

**Male *Usr* features:** boy, mike, ben, guy, mr, dad, jr, kid, tony, dog, lord, sir, omar, dude, man, big

**Female *BoW* features:** hubby, hubs, jewelry, sewing, mascara, fabulous, bf, softball, betcha, motherhood, perky, cozy, zumba, xox, cuddled, belieber, bridesmaid, anorexic, jammies, pad

**Female *Usr* features:** mrs, mom, jen, lady, wife, mary, joy, mama, pink, kim, diva, elle, woma, ms

Table 4: Examples of highly-weighted *BoW* (content) and *Usr* (username) features (in descending order of weight) in the *Bootstrapped* system for predicting user gender in Twitter.

Many recent papers have analyzed the language of social media users, along dimensions such as ethnicity (Eisenstein et al., 2011; Rao et al., 2011; Pennacchiotti and Popescu, 2011; Fink et al., 2012) time zone (Kiciman, 2010), political orientation (Rao et al., 2010; Pennacchiotti and Popescu, 2011) and gender (Rao et al., 2010; Burger et al., 2011; Van Durme, 2012).

**Class-Attribute Extraction** The idea of using simple patterns to extract useful semantic relations goes back to Hearst (1992) who focused on hyponyms. Hearst reports that she “tried applying this technique to meronymy (i.e., the part/whole relation), but without great success.” Berland and Charniak (1999) did have success using Hearst-style patterns for part-whole detection, which they attribute to their “very large corpus and the use of more refined statistical measures for ranking the output.” Girju et al. (2006) devised a supervised classification scheme for part/whole relation discovery that integrates the evidence from multiple patterns. These efforts focused exclusively on the *meronymy* relation as used in WordNet (Miller et al., 1990). Indeed, Berland and Charniak (1999) attempted to filter out attributes that were regarded as *qualities* (like *driveability*) rather than parts (like *steering wheels*) by removing words ending with the suffixes *-ness*, *-ing*, and *-ity*. In our work, such qualities are not filtered and are ultimately valuable in classification; for example, the attributes *peak fertility* and *loveliness* are highly

associated with females.

As subsequent research became more focused on applications, looser definitions of class attributes were adopted. Almuhareb and Poesio (2004) automatically mined class attributes that include parts, qualities, and those with an “agentive” or “telic” role with the class. Their extended set of attributes was shown to enable an improved representation of nouns for the purpose of clustering these nouns into semantic concepts. Tokunaga et al. (2005) define attributes as properties that can serve as focus words in questions about a target class; e.g. *director* is an attribute of a *movie* since one might ask, “Who is the *director* of this *movie*?” Another line of research has been motivated by the observation that much of Internet search consists of people looking for *values* of various class attributes (Bellare et al., 2007; Paşca and Van Durme, 2007; Paşca and Van Durme, 2008; Alfonseca et al., 2010). By knowing the attributes of different classes, search engines can better recognize that queries such as “altitude guadalajara” or “population guadalajara” are seeking *values* for a particular city’s “altitude” and “population” *attributes* (Paşca and Van Durme, 2007). Finally, note that Van Durme et al. (2008) compared instance-based and class-based patterns for broad-definition attribute extraction, and found both to be effective.

Of course, text-mining with custom-designed patterns is not the only way to extract class-attribute information. Experts can manually specify the attributes of entities, as in the WordNet project (Miller et al., 1990). Others have automatically extracted attribute relations from dictionary definitions (Richardson et al., 1998), structured online sources such as Wikipedia infoboxes, (Wu and Weld, 2007) and large-scale collections of high-quality tabular web data (Cafarella et al., 2008). Attribute extraction has also been viewed as a sub-component or special case of the information obtained by general-purpose knowledge extractors (Schubert, 2002; Pantel and Pennacchiotti, 2006).

**NLP Applications of Common-Sense Knowledge** The kind of information derived from class-attribute extraction is sometimes referred to as a type of *common-sense knowledge*. The need for computer programs to represent common-sense knowledge has been recognized since the work of McCarthy (1959). Lenat et al. (1990)

defines common sense as “human consensus reality knowledge: the facts and concepts that you and I know and which we each assume the other knows.”

While we are the first to exploit common-sense knowledge in user characterization, common sense has been applied to a range of other problems in natural language processing. In many ways WordNet can be regarded as a collection of common-sense relationships. WordNet has been applied in a myriad of NLP applications, including in seminal works on semantic-role labeling (Gildea and Jurafsky, 2002), coreference resolution (Soon et al., 2001) and spelling correction (Budanitsky and Hirst, 2006). Also, many approaches to the task of sentiment analysis “begin with a large lexicon of words marked with their prior polarity” (Wilson et al., 2009). Like our class-attribute associations, the common-sense knowledge that the word *cool* is positive while *unethical* is negative can be learned from associations in web-scale data (Turney, 2002). We might also view information about *synonyms* or *conceptually-similar words* as a kind of common-sense knowledge. In this perspective, our work is related to recent work that has extracted distributionally-similar words from web-scale data and applied this knowledge in tasks such as named-entity recognition (Lin and Wu, 2009) and dependency parsing (Täckström et al., 2012).

## 8 Conclusion

We have proposed, developed and successfully evaluated a novel approach to user characterization based on exploiting knowledge of user class attributes. The knowledge is obtained using a new algorithm that discovers *distinguishing* attributes of particular classes. Our approach to discovering distinguishing attributes represents a significant new direction for research in class-attribute extraction, and provides a valuable bridge between the fields of user characterization and lexical knowledge extraction.

We presented three effective techniques for leveraging this knowledge within the framework of supervised user characterization: rule-based post-processing, a learning-by-bootstrapping approach, and a stacking approach that integrates the predictions of the bootstrapped system into a system trained on annotated gold-standard training data. All techniques lead to significant improve-

ments over state-of-the-art supervised systems on the task of Twitter gender classification.

While our technique has advanced the state-of-the-art on this important task, our approach may prove even more useful on other tasks where training on thousands of gold-standard examples is not even an option. Currently we are exploring the prediction of finer-grained user roles, such as *student*, *waitress*, *parent*, and so forth, based on extensions to the process laid out here.

## References

- Enrique Alfonseca, Marius Paşca, and Enrique Robledo-Arnuncio. 2010. Acquisition of instance attributes via labeled and related instances. In *Proc. SIGIR*, pages 58–65.
- Abdulrahman Almuḥareb and Massimo Poesio. 2004. Attribute-based and value-based clustering: An evaluation. In *Proc. EMNLP*, pages 158–165.
- Kedar Bellare, Partha P. Talukdar, Giridhar Kumaran, Fernando Pereira, Mark Liberman, Andrew McCallum, and Mark Dredze. 2007. Lightly-Supervised Attribute Extraction. In *NIPS Workshop on Machine Learning for Web Search*.
- Shane Bergsma and Dekang Lin. 2006. Bootstrapping path-based pronoun resolution. In *Proc. Coling-ACL*, pages 33–40.
- Shane Bergsma, Paul McNamee, Mossaab Bagdouri, Clayton Fink, and Theresa Wilson. 2012. Language identification for creating language-specific Twitter collections. In *Proceedings of the Second Workshop on Language in Social Media*, pages 65–74.
- Shane Bergsma, Mark Dredze, Benjamin Van Durme, Theresa Wilson, and David Yarowsky. 2013. Broadly improving user classification via communication-based name and location clustering on twitter. In *Proc. NAACL*.
- Matthew Berland and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proc. ACL*, pages 57–64.
- Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.
- John D. Burger and John C. Henderson. 2006. An exploration of observable features related to blogger age. In *Proc. AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 15–20.
- John D. Burger, John Henderson, George Kim, and Guido Zarrella. 2011. Discriminating gender on Twitter. In *Proc. EMNLP*, pages 1301–1309.
- Michael J. Cafarella, Alon Y. Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. WebTables: exploring the power of tables on the web. *Proc. PVLDB*, 1(1):538–549.
- Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1).
- Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26.
- Jacob Eisenstein, Noah A. Smith, and Eric P. Xing. 2011. Discovering sociolinguistic associations with structured sparsity. In *Proc. ACL*, pages 1365–1374.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874.
- Clayton Fink, Jonathon Kopecky, Nathan Bos, and Max Thomas. 2012. Mapping the Twitterverse in the developing world: An analysis of social media use in Nigeria. In *Proc. International Conference on Social Computing, Behavioral Modeling, and Prediction*, pages 164–171.
- John L. Fischer. 1968. Social influences on the choice of a linguistic variant. *Word*, 14:47–56.
- Nikesh Garera and David Yarowsky. 2009. Modeling latent biographic attributes in conversational genres. In *Proc. ACL-IJCNLP*, pages 710–718.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288.
- Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. Coling*, pages 539–545.
- Emre Kiciman. 2010. Language differences and metadata features on Twitter. In *Proc. SIGIR 2010 Web N-gram Workshop*, pages 47–51.
- Moshe Koppel and Jonathan Schler. 2004. Authorship verification as a one-class classification problem. In *Proc. ICML*, pages 489–495.
- Moshe Koppel, Shlomo Argamon, and Anat Rachel Shimoni. 2002. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17(4):401–412.
- Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Determining an author’s native language by mining a text for errors. In *Proc. KDD*, pages 624–628.

- William Labov. 1972. *Sociolinguistic Patterns*. University of Pennsylvania Press.
- Douglas B. Lenat, R. V. Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd. 1990. CYC: toward programs with common sense. *Commun. ACM*, 33(8):30–49.
- Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proc. ACL-IJCNLP*, pages 1030–1038.
- Dekang Lin, Kenneth Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, and Sushant Narsale. 2010. New tools for web-scale N-grams. In *Proc. LREC*, pages 2221–2227.
- John McCarthy. 1959. Programs with common sense. In *Proc. Teddington Conference on the Mechanization of Thought Processes*, pages 75–91. London: Her Majesty’s Stationery Office.
- Ken McRae, George S. Cree, Mark S. Seidenberg, and Chris McNorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37(4):547–559.
- George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4).
- Arjun Mukherjee and Bing Liu. 2010. Improving gender classification of blog authors. In *Proc. EMNLP*, pages 207–217.
- Brendan O’Connor, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. 2010. From tweets to polls: Linking text sentiment to public opinion time series. In *Proc. ICWSM*, pages 122–129.
- Olutobi Owoputi, Brendan O’Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proc. of NAACL*.
- Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *Proc. Coling-ACL*, pages 113–120.
- Marius Paşca and Benjamin Van Durme. 2007. What you seek is what you get: extraction of class attributes from query logs. In *Proc. IJCAI*, pages 2832–2837.
- Marius Paşca and Benjamin Van Durme. 2008. Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. In *Proc. ACL-08: HLT*, pages 19–27.
- Michael Paul and Mark Dredze. 2011. You are what you tweet: Analyzing Twitter for public health. In *Proc. ICWSM*, pages 265–272.
- Marco Pennacchiotti and Ana-Maria Popescu. 2011. A machine learning approach to Twitter user classification. In *Proc. ICWSM*, pages 281–288.
- Xuan-Hieu Phan. 2006. CRFTagger: CRF English POS Tagger. [crftagger.sourceforge.net](http://crftagger.sourceforge.net).
- Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. 2010. Classifying latent user attributes in Twitter. In *Proc. International Workshop on Search and Mining User-Generated Contents*, pages 37–44.
- Delip Rao, Michael Paul, Clay Fink, David Yarowsky, Timothy Oates, and Glen Coppersmith. 2011. Hierarchical bayesian models for latent attribute detection in social media. In *Proc. ICWSM*, pages 598–601.
- Joseph Reisinger and Marius Paşca. 2009. Latent variable models of concept-attribute attachment. In *Proc. ACL-IJCNLP*, pages 620–628.
- Stephen D. Richardson, William B. Dolan, and Lucy Vanderwende. 1998. MindNet: Acquiring and structuring semantic information from text. In *Proc. ACL-Coling*, pages 1098–1102.
- Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W. Pennebaker. 2006. Effects of age and gender on blogging. In *Proc. AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 199–205.
- Lenhart Schubert. 2002. Can we derive general world knowledge from texts? In *Proc. HLT*, pages 84–87.
- Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34:1–47.
- Ian Soboroff, Dean McCullough, Jimmy Lin, Craig Macdonald, Iadh Ounis, and Richard McCreadie. 2012. Evaluating real-time search over tweets. In *Proc. ICWSM*.
- Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4).
- Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proc. NAACL-HLT*, pages 477–487.
- Kosuke Tokunaga, Jun’ichi Kazama, and Kentaro Torisawa. 2005. Automatic discovery of attribute words from web documents. In *Proc. IJCNLP*, pages 106–118.

- Peter D. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proc. ACL*, pages 417–424.
- Benjamin Van Durme, Ting Qian, and Lenhart Schubert. 2008. Class-driven attribute extraction. In *Proc. Coling*, pages 921–928.
- Benjamin Van Durme. 2012. Streaming analysis of discourse participants. In *Proc. EMNLP-CoNLL*, pages 48–58.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2009. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational Linguistics*, 35(3):399–433.
- Fei Wu and Daniel S. Weld. 2007. Autonomously semantifying Wikipedia. In *Proc. CIKM*, pages 41–50.
- Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using “annotator rationales” to improve machine learning for text categorization. In *Proc. NAACL-HLT*.

# The Impact of Topic Bias on Quality Flaw Prediction in Wikipedia

Oliver Ferschke<sup>†</sup>, Iryna Gurevych<sup>†‡</sup> and Marc Rittberger<sup>‡</sup>

<sup>†</sup> Ubiquitous Knowledge Processing Lab

Department of Computer Science, Technische Universität Darmstadt

<sup>‡</sup> Information Center for Education

German Institute for Educational Research and Educational Information

<http://www.ukp.tu-darmstadt.de>

## Abstract

With the increasing amount of user generated reference texts in the web, automatic quality assessment has become a key challenge. However, only a small amount of annotated data is available for training quality assessment systems. Wikipedia contains a large amount of texts annotated with cleanup templates which identify quality flaws. We show that the distribution of these labels is topically biased, since they cannot be applied freely to any arbitrary article. We argue that it is necessary to consider the topical restrictions of each label in order to avoid a sampling bias that results in a skewed classifier and overly optimistic evaluation results. We factor out the topic bias by extracting reliable training instances from the revision history which have a topic distribution similar to the labeled articles. This approach better reflects the situation a classifier would face in a real-life application.

## 1 Introduction

User generated content is the main driving force of the increasingly social web. Blogs, wikis and forums make up a large amount of the daily information consumed by web users. The main properties of user generated content are a low publication threshold and little or no editorial control, which leads to a high variance in quality. In order to navigate through large repositories of information efficiently and safely, users need a way to quickly assess the quality of the content. Automatic quality assessment has therefore become a key application in today's information society. However, there is a lack of training data annotated with fine-grained quality information.

Wikipedia, the largest encyclopedia on the web,

contains so-called cleanup templates, which constitute a sophisticated system of user generated labels that mark quality problems in articles. Recently, these cleanup templates have been used for automatically identifying articles with particular quality flaws in order to support Wikipedia's quality assurance process in Wikipedia. In a shared task (Anderka and Stein, 2012b), several systems have shown that it is possible to identify the ten most frequent quality flaws with high recall and fair precision.

However, quality flaw detection based on cleanup template recognition suffers from a topic bias that is well known from other text classification applications such as authorship attribution or genre identification. We discovered that cleanup templates have implicit topical restrictions, i.e. they cannot be applied to any arbitrary article. As a consequence, corpora of flawed articles based on these templates are biased towards particular topics. We argue that it is therefore not sufficient for evaluating a quality flaw prediction systems to measure how well they can separate (topically restricted) flawed articles from a set of random outliers. It is rather necessary to determine reliable negative instances with a similar topic distribution as the set of positive instances in order to factor out the sampling bias. Related studies (Brooke and Hirst, 2011) have proven that topic bias is a confounding factor that results in misleading cross-validated performance while allowing only near chance performance in practical applications.

We present an approach for factoring out the bias from quality flaw corpora by mining reliable negative instances for each flaw from the article revision history. Furthermore, we employ the article revision history to extract reliable positive training instances by using the version of each article at the time it has first been identified as flawed. This way, we avoid including articles with outdated cleanup templates, a frequent phe-

nomenon that can occur when a template is not removed after fixing a problem in an article. In our experiments, we focus on neutrality and style flaws, since they are of particular high importance within the Wikipedia community (Stvilia et al., 2008; Ferschke et al., 2012a) and are recognized beyond Wikipedia in applications such as uncertainty recognition (Szarvas et al., 2012) and hedge detection (Farkas et al., 2010).

## 2 Related Work

Topic bias is a known problem in text classification. Mikros and Argiri (2007) investigate the topic influence in authorship attribution. They found that even simple stylometric features, such as sentence and token length, readability measures or word length distributions show considerable correlations with topic. They argue that many features that were largely considered to be topic neutral are in fact topic-dependent variables. Consequently, results obtained on multitopic corpora are prone to be biased by the correlation of authors with specific topics. Therefore, several authors introduce topic-controlled corpora for applications such as author identification (Koppel and Schler, 2003; Luyckx and Daelemans, 2005) or genre detection (Finn and Kushmerick, 2006).

Brooke and Hirst (2011) measure the topic bias in the *International Corpus of Learner English* and found that it causes a substantial skew in classifiers for native language detection. In accordance with Mikros et al., the authors found that even non-lexicalized meta features, such as vocabulary size or length statistics, depend on topics and cause cross-validated performance evaluations to be unrealistically high. In a practical setting, these biased classifiers hardly exceed chance performance.

As already noted above, a similar kind of topic bias negatively influences quality flaw detection in Wikipedia. Anderka et al. (2012) automatically identify quality flaws by predicting the cleanup templates in unseen articles with a one-class classification approach. Based on this work, a competition on quality flaw prediction has been established (Anderka and Stein, 2012b). The winning team of the inaugural edition of the task was able to detect the ten most common quality flaws with an average  $F_1$ -Score of 0.81 using a PU learning approach (Ferretti et al., 2012). With a binary classification approach, Ferschke et

al. (2012b) achieved an average  $F_1$ -Score of 0.80, while reaching a higher precision than the winning team.

A closer examination of the aforementioned quality flaw detection systems reveals a systematic sampling bias in the training data, which leads to an overly optimistic performance evaluation and classifiers that are biased towards particular article topics. Our approach factors out the topic bias from the training data by mining topically controlled training instances from the Wikipedia revision history. The results show that flaw detection is a much harder problem in a real-life scenario.

## 3 Quality Flaws and Flaw Recognition in Wikipedia

Quality standards in Wikipedia are mainly defined by the *featured article criteria*<sup>1</sup> and the *Wikipedia Manual of Style*<sup>2</sup>. These policies define the characteristics excellent articles have to exhibit. Other sets of quality criteria are adaptations or relaxations of these standards, such as the *good article criteria* or the quality grading schemes of individual interest groups in Wikipedia.

In this work, we focus on quality flaws regarding neutrality and style problems. We chose these categories due to their high importance within the Wikipedia community (Stvilia et al., 2008; Ferschke et al., 2012a) and due to their relevance to content outside of Wikipedia, such as blogs or online news articles. According to the Wikipedia policies<sup>3</sup>, an article has to be written from a neutral point of view. Thus, authors must avoid stating opinions and seriously contested assertions as facts, avoid presenting uncontested factual assertions as mere opinions, prefer nonjudgmental language and indicate the relative prominence of opposing views. Furthermore, authors have to adhere to the stylistic guidelines defined in the Manual of Style. While this subsumes a broad range of issues such as formatting and article structure, we focus on the style of writing and disregard mere structural properties.

Any articles that violate these criteria can be marked with cleanup templates<sup>4</sup> to indicate their need for improvement. These templates can thus be regarded as proxies for quality flaws in Wikipedia.

<sup>1</sup><http://en.wikipedia.org/wiki/WP:FACR>

<sup>2</sup><http://en.wikipedia.org/wiki/WP:STYLE>

<sup>3</sup><http://en.wikipedia.org/wiki/WP:NPOV>

<sup>4</sup><http://en.wikipedia.org/wiki/WP:TM#Cleanup>

	Flaw	Description	Articles	Templates
Neutrality	<b>Advert</b>	The article appears to be written like an advertisement and is thus not neutral	7,332	2
	<b>POV</b>	The neutrality of this article is disputed	5,086	10
	<b>Globalize</b>	The article may not represent a worldwide view of the subject	1,609	1
	<b>Peacock</b>	The article may contain wording that merely promotes the subject without imparting verifiable information	1,195	1
	<b>Weasel</b>	The article contains vague phrasing that often accompanies biased or unverifiable information	704	4
Style	<b>Tone</b>	The tone of the article is not encyclopedic according to the Wikipedia Manual of Style	4,563	6
	<b>In-universe</b>	The article describes a work or element of fiction in a primarily in-universe style <sup>a</sup>	2,227	1
	<b>Copy-edit</b>	The article requires copy editing for grammar, style, cohesion, tone, or spelling	1,954	6
	<b>Trivia</b>	Contains lists of miscellaneous information	1,282	2
	<b>Essay-like</b>	The article is written like a personal reflection or essay	1,244	1
	<b>Confusing</b>	The article may be confusing or unclear to readers	1,084	1
	<b>Technical</b>	The article may be too technical for most readers to understand	690	2

<sup>a</sup> According to the Wikipedia Manual of Style, an in-universe perspective describes the article subject matter from the perspective of characters within a fictional universe as if it were real.

Table 1: Neutrality and style flaw corpora used in this work

**Template Clusters** Since several cleanup templates might represent different manifestations of the same quality flaw, there is a *1 to n* relationship between quality flaws and cleanup templates. For instance, the templates `pov-check`<sup>5</sup>, `pov`<sup>6</sup> and `npov language`<sup>7</sup> can all be mapped to the same flaw concerning the neutral point of view of an article. This aggregation of cleanup templates into flaw-clusters is a subjective task. It is not always clear whether a particular template refers to an existing flaw or should be regarded as a separate class. Too many clusters will cause definition overlaps (i.e. similar cleanup templates are assigned to different clusters), while too few clusters will result in unclear flaw definitions, since each flaw receives a wide range of possible manifestations.

**Template Scope** Another important aspect to be considered is the difference in the scope which cleanup templates can have. *Inline-templates* are placed directly in the text and refer to the sentence or paragraph they are placed in. Templates with a *section* parameter, refer to the section they are placed in. The majority of templates, however, refer to a whole page. The consideration of the template scope is of particular importance for quality flaw recognition problems. For example, the presence of a cleanup template which marks a single section as *not notable* does not entail that the whole article is not notable.

<sup>5</sup>The article has been nominated for a neutrality check

<sup>6</sup>The neutrality of the article is disputed

<sup>7</sup>The article contains a non-neutral style of writing

**Topical Restriction** A final aspect that has not been taken into account by related work is that many cleanup templates have restrictions concerning the pages they may be applied to. A hard restriction is the page type (or namespace) a template might be used in. For example, some templates can only be used in articles while others can only be applied to discussion pages. This is usually enforced by maintenance scripts running on the Wikimedia servers. A soft restriction, on the other hand, are the topics of the articles a template can be used in. Many cleanup templates can only be applied to articles from certain subject areas. An example with a particularly obvious restriction is the template *in-universe* (see Table 1), which should only be applied to articles about fiction. This topical restriction is neither explicitly defined nor automatically enforced, but it plays an important role in the quality flaw recognition task, as the remainder of this paper will show. While flaws merely concerning the structural or linguistic properties of an article are less restricted to individual topics, they are still affected by a certain degree of *topical preference*. Many subject areas in Wikipedia are organized in *WikiProjects*<sup>8</sup>, which have their own ways of reviewing and ensuring quality within their topical scope. Depending on the quality assurance processes established in a WikiProject, different importance is given to individual types of flaws. Thus, the distribution of cleanup templates regarding structural or grammatical flaws is also biased towards certain topics.

<sup>8</sup><http://en.wikipedia.org/wiki/WP:PROJ>

We will henceforth subsume the concept of topical preference under the term topical restriction.

**Quality Flaw Recognition** Based on the above definition of quality flaws, we define the quality flaw recognition task similar to Anderka et al. (2012) as follows: Given a sample of articles in which each article has been tagged with any cleanup template  $\tau_i$  from a specific template cluster  $T_f$  thus marking all articles in the sample with a quality flaw  $f$ , it has to be decided whether or not an unseen article suffers from  $f$ .

#### 4 Data Selection and Corpus Creation

For creating our corpora, we start with selecting all cleanup templates listed under the categories *neutrality* and *style* in the typology of cleanup templates provided by Anderka and Stein (2012a). Each of the selected templates serves as the nucleus of a template cluster that potentially represents a quality flaw. To each cluster, we add all templates that are synonymous to the nucleus. The synonyms are listed in the template description under *redirects* or *shortcuts*. Then we iteratively add all synonyms of the newly added template until no more redirects can be found. Furthermore, we manually inspect the lists of similar templates in the *see also* sections of the template descriptions and include all templates that refer to the same concept as the other templates in the cluster. As mentioned earlier, this is a subjective task and largely depends on the desired granularity of the flaw definitions. We finally merge semantically similar template clusters to avoid too fine grained flaw distinctions.

As a result, we obtain a total number of 94 template clusters representing 60 style flaws and 34 neutrality flaws. From each of these clusters, we remove templates with inline or section scope due to the reasons outlined in Section 3. We also remove all templates that are restricted to pages other than articles (e.g. discussion or user pages).

We use the Java Wikipedia Library (Zesch et al., 2008) to extract all articles marked with the selected templates. We only regard flaws with at least 500 affected articles in the snapshot of the English Wikipedia from January 4, 2012. Table 1 lists the final sets of flaws used in this work. For each flaw, the nucleus of the template cluster is provided along with a description, the number of affected articles, and the cluster size. We make the corpora freely available for down-

Flaw	$\kappa$	$F_1$
Advert	.60	.80
Confusing	.60	.80
Copy-edit	.00	.50
Essay-like	.60	.80
Globalize:	.60	.80
In-universe	.80	.90
Peacock	.70	.84
POV	.60	.80
Technical	.90	.95
Tone	.40	.70
Trivia	.20	.60
Weasel	.50	.74

Table 2: Agreement of human annotator with gold standard

load under <http://www.ukp.tu-darmstadt.de/data/wiki-flaws/>.

#### Agreement with Human Rater

Quality flaw detection in Wikipedia is based on the assumption that cleanup templates are valid markers of quality flaws. In order to test the reliability of these user assigned templates as quality flaw markers, we carried out an annotation study in which a human annotator was asked to perform the binary flaw detection task manually. Even though the human performance does not necessarily provide an upper boundary for the automatic classification task, it gives insights into potentially problematic cases and ill-defined annotations. The annotator was provided with the template definitions from the respective template information page as instructions. For each of the 12 article scope flaws, we extracted the plain text of 10 random flawed articles and 10 random untagged articles. The annotator had to decide for each flaw individually whether a given text belonged to a flawed article or not. She was not informed about the ratio of flawed to untagged articles.

Table 2 lists the chance corrected agreement (Cohen’s  $\kappa$ ) along with the  $F_1$  performance of the human annotations against the gold standard corpus. The templates *copy-edit* and *trivia* yielded the lowest performance in the study. Even though *copy-edit* templates are assigned to whole articles, they refer to grammatical and stylistic problems of relatively small portions of the text. This increases the risk of overlooking a problematic span of text, especially in longer articles. The *trivia* template, on the other hand, designates sections that contain miscellaneous information that are not well integrated in the article. Upon manual inspection, we found a wide range of possible manifestations of

this flaw ranging from an agglomeration of incoherent factoids to well-structured sections that did not exactly match the focus of the article, which is the main reason for the low agreement.

## 5 Selection of Reliable Training Instances

Independent from the classification approach used to identify flawed articles, reliable training data is the most important prerequisite for good predictions. On the one hand, we need a set of examples that reliably represent a particular flaw, while on the other hand, we need counterexamples which reliably represent articles that do not suffer from the same flaw. The latter aspect is most important for discriminative classification approaches, since they rely on negative instances for training the classifier. However, reliable negative instances are also important for one-class classification approaches, since it is only for the counterexamples (or outliers) that the performance of one-class classifiers can be sufficiently evaluated. It is furthermore important that the positive and the negative instances do not differ *systematically* in any respect other than the presence or absence of the respective flaws, since any systematic difference will bias the classifier. In this context, the topical restrictions of cleanup templates have to be taken into account. In the following, we describe our approach to extracting reliable training instances from the quality flaw corpora.

### 5.1 Reliable Positives

In previous work, the latest available versions of flawed articles have been used as positive training instances. However, we found upon manual inspection of the data that a substantial number of articles has been significantly edited between the time  $t_\tau$ , at which the template was first assigned, and the time  $t_e$ , at which the articles have been extracted. Using the latest version at time  $t_e$  can thus include articles in which the respective flaw has already been fixed without removing the cleanup template. Therefore, we use the revision of the article at time  $t_\tau$  to assure that the flaw is still present in the training instance.

We use the Wikipedia Revision Toolkit (Ferschke et al., 2011), an enhancement of the Java Wikipedia Library, to gain access to the revision history of each article. For every article in the corpus of positive examples for flaw  $f$  that is marked

with template  $\tau \in T_f$ , we backtrack the revision history chronologically, until we find the first revision  $r_{t_{\tau-1}}$  that is not tagged with  $\tau$ . We then add the succeeding revision  $r_{t_\tau}$  to the corpus of reliable positives for flaw  $f$ . In Section 6, we show that the classification performance improves for most flaws when using reliable positives instead of the latest available article versions.

### 5.2 Reliable Negatives and Topical Restriction

A central problem of the quality flaw recognition approach is the fact that there are no articles available that are tagged to not contain a particular quality problem. So far, two solutions to this issue have been proposed in related work. Anderka et al. (2012) tackle the problem with a one-class classifier that is trained on the positive instances alone thus eradicating the need for negative instances in the training phase. However, in order to evaluate the classifier, a set of outliers is needed. The authors circumvent this issue by evaluating their classifiers on a set of random untagged instances and a set of featured articles and argue that the actual performance of predicting the quality flaws lies between the two.

Ferretti et al. (2012) follow a two step classification approach (PU learning) that first uses a Naive Bayes classifier trained on positive instances and random untagged articles to pre-classify the data. In a second phase, they use the negatives identified by the Naive Bayes classifier to train a Support Vector Machine that produces the final predictions. Even though the Naive Bayes classifier was supposed to identify reliable negatives, the authors found no significant improvement over a random selection of negative instances, which effectively renders the PU learning approach redundant.

None of the above approaches consider the issue of topical restriction mentioned in Section 3, which introduces a systematic bias to the data. Both approaches sample random negative instances  $A_{rnd}$  for any given set of flawed articles  $A_f$  from a set of untagged articles  $A_u$  (see Fig. 1a). In order to factor out the article topics as a major characteristic for distinguishing flawed articles from the set of outliers, reliable negative instances  $A_{rel}$  have to be sampled from the restricted topic set  $A_{topic}$  that contains articles with a topic distribution similar to the flawed articles in  $A_f$  (see Fig. 1b). This will avoid the systematic bias and

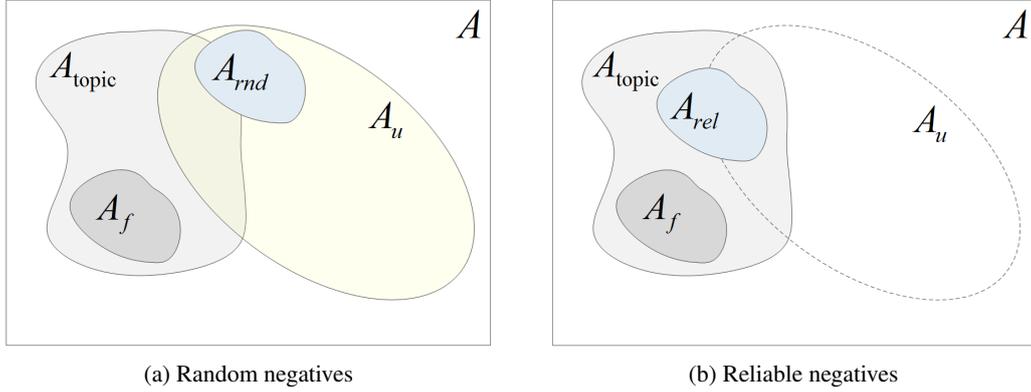


Figure 1: Sampling of negative instances for a given set of flawed articles ( $A_f$ ). Random negatives ( $A_{rnd}$ ) are sampled from articles without any cleanup templates ( $A_u$ ). Reliable negatives ( $A_{rel}$ ) are sampled from the set of articles ( $A_{topic}$ ) with the same topic distribution as  $A_f$

result in a more realistic performance evaluation.

In the following, we present our approach to extracting reliable negative training instances that conform with the topical restrictions of the cleanup templates. Without loss of generality, we assume that an article, from which a cleanup template  $\tau \in T_f$  is deleted at a point in time  $d_\tau$ , the article no longer suffers from flaw  $f$  at that point in time. Thus, the revision  $r_{d_\tau}$  is a reliable *negative instance* for the flaw  $f$ . Additionally, since the article was once tagged with  $\tau \in T_f$ , it belongs to the the same restricted topic set  $A_{topic}$  as the positive instances for flaw  $f$ .

We use the *Apache Hadoop*<sup>9</sup> framework and *WikiHadoop*<sup>10</sup>, an input format for Wikipedia XML dumps, for crawling the whole revision history of the English Wikipedia on a compute cluster. WikiHadoop allows each Hadoop mapper to receive adjacent revision pairs, which makes it possible to compare the changes made from one revision to the next. For every template  $\tau \in T_f$ , we extract all adjacent revision pairs  $(r_{d_\tau-1}, r_{d_\tau})$ , in which the first revision contains  $\tau$  and the second does not contain  $\tau$ . Since there are occasions in which a template is replaced by another template from the same cluster, we ensure that  $r_{d_\tau}$  does also not contain any other template from cluster  $T_f$  before we finally add the revision to the set of reliable negatives for flaw  $f$ .

In the remainder of this section, we evaluate the topical similarity between the positive and the negative set of articles for each flaw using both our method and the original approach. In Wikipedia,

the topic of an article is captured by the categories assigned to it. In order to compare two sets of articles with respect to their topical similarity, we represent each article set as a category frequency vector. Formally, we calculate for each set the vector  $\vec{C} = (w_{c_1}, w_{c_2}, \dots, w_{c_n})$  with  $w_{c_i}$  being the weight of category  $c_i$ , i.e. the number of times it occurs in the set, and  $n$  being the total number of categories in Wikipedia. We can then estimate the topical similarity of two article sets by calculating the cosine similarity of their category frequency vectors  $\vec{C}_1 := A$  and  $\vec{C}_2 := B$  as

$$\text{sim}(A, B) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$

Table 3 gives an overview of the similarity scores between each positive training set and the corresponding reliable negative set as well as between each positive set and a random set of untagged articles. We can see that the topics of articles in the positive training sets are highly similar to the topics of the corresponding reliable negative articles while they show little similarity to the articles in the random set. This implies that the systematic bias introduced by the topical restriction has largely been eradicated by our approach.

Individual flaws have differently strong topical restrictions. The strength of this restriction depends on the size of  $A_{topic}$ . That is, a flaw such as *in-universe* is restricted to a very narrow selection of articles, while a flaw such as *copy edit* can be applied to most articles and rather shows a topical preference due to reasons outlined in Section 3. It

<sup>9</sup><http://hadoop.apache.org>

<sup>10</sup><https://github.com/whym/wikihadoop>

Flaw	Cosine Similarity	
	$(A_f, A_{rel})$	$(A_f, A_{rnd})$
Advert	.996	.118
Confusing	.996	.084
Copy-edit	.993	.197
Essay-like	.996	.132
Globalize	.992	.023
In-universe	.996	.014
Peacock	.995	.310
POV	.994	.252
Technical	.995	.018
Tone	.996	.228
Trivia	.980	.184
Weasel	.976	.252

Table 3: Cosine similarity scores between the category frequency vectors of the flawed article sets and the respective random or reliable negatives

is therefore to be expected that that flaws with a small  $A_{topic}$  are more prone to the topic bias.

## 6 Experiments

In the following, we describe our system architecture and the setup of our experiments. Our system for quality flaw detection follows the approach by Ferschke et al. (2012b), since it has been particularly designed as a modular system based on the Unstructured Information Management Architecture<sup>11</sup>, which makes it easy to extend. Instead of using *Mallet* (McCallum, 2002) as a machine learning toolkit, we employ the *Weka Data Mining Software* (Hall et al., 2009) for classification, since it offers a wider range of state-of-the-art machine learning algorithms. For each of the 12 quality flaws, we employ three different dataset configurations. The BASE configuration uses the newest version of each flawed article as positive instances and a random set of untagged articles as negative instances. The RELP configuration uses reliable positives, as described in Section 5.1, in combination with random outliers. Finally, the RELALL configuration employs reliable positives in combination with the respective reliable negatives as described in Section 5.2.

### Features

An extensive survey of features for quality flaw recognition has been provided by Anderka et al. (2012). We selected a subset of these features for our experiments and grouped them into four feature sets in order to determine how well different combinations of features perform in the task.

<sup>11</sup><http://uima.apache.org>

Category	Feature type	NONGRAM			GRAM			NOWIKI			ALL		
Lexical	Article ngrams				•	•	•						
	Info to noise ratio				•		•						
Network	# External links				•								
	# Outlinks				•								
	# Outlinks per sentence				•								
	# Language links				•								
References	Has reference list				•								
	# References				•								
	# References per sentence				•								
	# Revisions				•								
Revision	# Unique contributors				•								
	# Empty sections				•								
	Mean section size				•								
	# Sections				•								
Structure	# Lists				•								
	Question rate				•		•		•				
	ARI				•		•		•				
	Coleman-Liau				•		•		•				
Readability	Flesch				•		•		•				
	Flesch-Kincaid				•		•		•				
	Gunning Fog				•		•		•				
	Lix				•		•		•				
Named Entity	SMOG-Grading				•		•		•				
	# Person entities*				•		•		•				
	# Organization entities*				•		•		•				
	# Location entities*				•		•		•				
Misc	# Characters				•		•		•				
	# Sentences				•		•		•				
	# Tokens				•		•		•				
	Average sentence length				•		•		•				
	Article lead length				•				•				
	Lead to article ratio				•				•				
	# Discussions				•				•				

\* newly introduced feature  
# number of instances

Table 4: Feature sets used in the experiments

Table 4 lists all feature types used in our experiments.

Since the feature space becomes large due to the ngram features, we prune it in two steps. First, we filter the ngrams according to their document frequency in the training corpus. We discard all ngrams that occur in less than  $x\%$  and more than  $y\%$  of all documents. Several values for  $x$  and  $y$  have been evaluated in parameter tuning experiments. The best results have been achieved with  $x=2$  and  $y=90$ . In a second step, we apply the Information Gain feature selection approach (Mitchell, 1997) to the remaining set to determine the most useful features.

### Learning Algorithms

We evaluated several learning algorithms from the Weka toolkit with respect to their performance on

Algorithm	Average $F_1$
<b>SVM RBF Kernel</b>	<b>0.82</b>
AdaBoost (decision stumps)	0.80
SVM Poly Kernel	0.79
RBF Network	0.78
SVM Linear Kernel	0.77
SVM PUK Kernel	0.76
J48	0.75
Naive Bayes	0.72
MultiBoostAB (decision stumps)	0.71
Logistic Regression	0.60
LibSVM One Class	0.67

Table 5: Average  $F_1$ -scores over all flaws on RELP using all features

the quality flaw recognition task. Table 5 shows the average  $F_1$ -score of each algorithm on the RELP dataset using all features. The performance has been evaluated with 10-fold cross validation on 2,000 documents split equally into positive and negative instances. One class classifiers are trained on the positive instances alone. We determined the best parameters for each algorithms in a parameter optimization run and list the results of the best configuration.

Overall, Support Vector Machines with RBF kernels yielded the best average results and outperformed the other algorithms on every flaw. We used a sequential minimal optimization (SMO) algorithm (Platt, 1998) to train the SVMs and used different  $\gamma$ -values for the RBF kernel function. In contrast to Ferretti et al. (2012), we did not see significant improvements when optimizing  $\gamma$  for each individual flaw, so we determined one best setting for each dataset. Since SVMs with RBF kernels are a special case of RBF networks that fit a single basis function to the data, we also used general RBF networks that can employ multiple basis functions, but we did not achieve better results with that approach.

One-class classification, as proposed by Anderka et al. (2012), did not perform well within our setup. Even though we used an out-of-the-box one class classifier, we achieve similar results as Anderka et al. in their pessimistic setting, which best resembles our configuration. However, the performance still lacks behind the other approaches in our experiments. The best performing algorithm reported by Ferschke et al. (2012b), AdaBoost with decision stumps as a weak learner, showed the second best results in our experiments.

## 7 Evaluation and Discussion

The SVMs achieve a similar cross-validated performance on all feature sets containing ngrams, showing only minor improvements for individual flaws when adding non-lexical features. This suggests that the classifiers largely depend on the ngrams and that other features do not contribute significantly to the classification performance. While structural quality flaws can be well captured by special purpose features or intentional modeling, as related work has shown, more subtle content flaws such as the neutrality and style flaws are mainly captured by the wording itself. Textual features beyond the ngram level, such as syntactic and semantic qualities of the text, could further improve the classification performance of these flaws and should be addressed in future work. Table 6 shows the performance of the SVMs with RBF kernel<sup>12</sup> on each dataset using the *NGRAM* feature set. The average performance based on *NOWIKI* is slightly lower while using *ALL* features results in slightly higher average  $F_1$ -scores. However, the differences are not statistically significant and thus omitted. Classifiers using the *NONGRAM* feature set achieved average  $F_1$ -scores below 0.50 on all datasets. The results have been obtained by 10-fold cross validation on 2,000 documents per flaw.

The classifiers trained on reliable positives and random untagged articles (RELP) outperform the respective classifiers based on the BASE dataset for most flaws. This confirms our original hypothesis that using the appropriate revision of each tagged article is superior to using the latest available version from the dump. The performance on the RELALL dataset, in which the topic bias has been factored out, yields lower  $F_1$ -scores than the two other approaches. Flaws that are restricted to a very narrow set of topics (i.e.  $A_{topic}$  in Fig. 1b is small), such as the *in-universe* flaw, show the biggest drop in performance. Since the topic bias plays a major role in the quality flaw detection task, as we have shown earlier, the topic-controlled classifier cannot take advantage of the topic information, while the classifiers trained on the other corpora can make use of these characteristic as the most discriminative features. In the RELALL setting, however, the differences between the positive and negative instances are largely determined by the flaws alone. Classifiers trained on

<sup>12</sup> $\gamma=0.01$  for BASE,RELp and  $\gamma=0.001$  for RELALL

such a dataset therefore come closer to recognizing the actual quality flaws, which makes them more useful in a practical setting despite lower cross-validated scores.

In addition to cross-validation, we performed a cross-corpus evaluation of the classifiers for each flaw. Therefore, we evaluated the performance of the unbiased classifiers (trained on RELALL) on the biased data (RELP) and vice versa. Hereby, the positive training and test instances remain the same in both settings, while the unbiased data contains negative instances sampled from  $A_{rel}$  and the unbiased data from  $A_{rnd}$  (see Figure 1). With the *NGRAM* feature set, the reliable classifiers outperformed the unreliable classifiers on all flaws that can be well identified with lexical cues, such as *Advert* or *Technical*. In the biased case, we found both topic related and flaw specific ngrams among the most highly ranked ngram features. In the unbiased case, most of the informative ngrams were flaw specific expressions. Consequently, biased classifiers fail on the unbiased dataset in which the positive and negative class are sampled from the same topics, which renders the highly ranked topic ngrams unusable. Flaws that do not largely rely on lexical cues, however, cannot be predicted more reliably with the unbiased classifier. This means that additional features are needed to describe these flaw. We tested this hypothesis by using the full feature set *ALL* and saw a substantial improvement on the side of the unbiased classifier, while the performance of the biased classifier remained unchanged.

A direct comparison of our results to related work is difficult, since neutrality and style flaws have not been targeted before in a similar manner. However, the *Advert* flaw was also part of the ten flaw types in the PAN Quality Flaw Recognition Task (Anderka and Stein, 2012b). The best system achieved an  $F_1$  score of 0.839, which is just below the results of our system on the BASE dataset, which is similar to the PAN setup.

## 8 Conclusions

We showed that text classification based on Wikipedia cleanup templates is prone to a topic bias which causes skewed classifiers and overly optimistic cross-validated evaluation results. This bias is known from other text classification applications, such as authorship attribution, genre detection and native language detection. We demon-

Flaw	BASE	RELP	RELALL
Advert	.86	.88	.75
Confusing	.76	.80	.70
Copy edit	.81	.73	.72
Essay-like	.79	.83	.64
Globalize	.85	.87	.69
In-universe	.96	.96	.69
Peacock	.77	.82	.69
POV	.75	.80	.71
Technical	.87	.88	.67
Tone	.70	.79	.69
Trivia	.72	.77	.70
Weasel	.69	.77	.72
$\emptyset$	.79	.83	.70

Table 6:  $F_1$  scores for the 10-fold cross validation of the SVMs with RBF kernel on all datasets using *NGRAM* features

strated how to avoid the topic bias when creating quality flaw corpora. Unbiased corpora are not only necessary for training unbiased classifiers, they are also invaluable resources for gaining a deeper understanding of the linguistic properties of the flaws. Unbiased classifiers reflect much better the performance of quality flaw recognition “in the wild”, because they detect actual flawed articles rather than identifying the articles that are prone to certain quality due to their topic or subject matter. In our experiments, we presented a system for identifying Wikipedia articles with style and neutrality flaws, a novel category of quality problems that is of particular importance within and outside of Wikipedia. We showed that selecting a reliable set of positive training instances mined from the revision history improves the classification performance. In future work, we aim to extend our quality flaw detection system to not only find articles that contain a particular flaw, but also to identify the flaws within the articles, which can be achieved by leveraging the positional information of in-line cleanup templates.

## Acknowledgments

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806, and by the Hessian research excellence program “Landes-Offensive zur Entwicklung Wissenschaftlich-Ökonomischer Exzellenz” (*LOEWE*) as part of the research center “Digital Humanities”.

## References

- Maik Anderka and Benno Stein. 2012a. A Breakdown of Quality Flaws in Wikipedia. In *2nd Joint WICOW/AIRWeb Workshop on Web Quality*, pages 11–18, Lyon, France.
- Maik Anderka and Benno Stein. 2012b. Overview of the 1st International Competition on Quality Flaw Prediction in Wikipedia. In *CLEF 2012 Evaluation Labs and Workshop – Working Notes Papers*.
- Maik Anderka, Benno Stein, and Nedim Lipka. 2012. Predicting Quality Flaws in User-generated Content: The Case of Wikipedia. In *35th International ACM Conference on Research and Development in Information Retrieval*, Portland, OR, USA.
- Julian Brooke and Graeme Hirst. 2011. Native language detection with ‘cheap’ learner corpora. In *Learner Corpus Research 2011 (LCR 2011)*.
- Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The CoNLL-2010 shared task: learning to detect hedges and their scope in natural language text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning, CoNLL ’10: Shared Task*, pages 1–12, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Edgardo Ferretti, Donato Hernández Fusilier, Rafael Guzmán-Cabrera, Manuel Montes y Gómez, Marcelo Errecalde, and Paolo Rosso. 2012. On the Use of PU Learning for Quality Flaw Prediction in Wikipedia. In *CLEF 2012 Evaluation Labs and Workshop – Working Notes Papers*.
- Oliver Ferschke, Torsten Zesch, and Iryna Gurevych. 2011. Wikipedia Revision Toolkit: Efficiently Accessing Wikipedia’s Edit History. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. System Demonstrations*, pages 97–102, Portland, OR, USA.
- Oliver Ferschke, Iryna Gurevych, and Yevgen Chebotar. 2012a. Behind the Article: Recognizing Dialog Acts in Wikipedia Talk Pages. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 777–786, Avignon, France.
- Oliver Ferschke, Iryna Gurevych, and Marc Rittberger. 2012b. FlawFinder: A Modular System for Predicting Quality Flaws in Wikipedia. In *CLEF 2012 Evaluation Labs and Workshop – Working Notes Papers*, Rome, Italy.
- Aidan Finn and Nicholas Kushmerick. 2006. Learning to classify documents according to genre. *Journal of the American Society for Information Science and Technology*, 57(11):1506–1518.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18.
- Moshe Koppel and Jonathan Schler. 2003. Exploiting stylistic idiosyncrasies for authorship attribution. In *Workshop on Computational Approaches to Style Analysis and Synthesis*, pages 69–72.
- K. Luyckx and W. Daelemans. 2005. Shallow text analysis and machine learning for authorship attribution. In *Proceedings of the Fifteenth Meeting of Computational Linguistics in the Netherlands (CLIN 2004)*, pages 149–160.
- Andrew Kachites McCallum. 2002. MALLET: A Machine Learning for Language Toolkit.
- George K. Mikros and Eleni K. Argiri. 2007. Investigating topic influence in authorship attribution. In *Proceedings of the SIGIR 2007 International Workshop on Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection, PAN 2007*, Amsterdam, Netherlands.
- Thomas Mitchell. 1997. *Machine Learning*. McGraw-Hill Education, New York, NY, USA, 1st edition.
- John C Platt. 1998. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Learning*, pages 185–208, Cambridge, MA, USA.
- Besiki Stvilia, Michael B. Twidale, Linda C. Smith, and Les Gasser. 2008. Information Quality Work Organization in Wikipedia. *Journal of the American Society for Information Science and Technology*, 59(6):983–1001.
- György Szarvas, Veronika Vincze, Richárd Farkas, György Móra, and Iryna Gurevych. 2012. Cross-genre and cross-domain detection of semantic uncertainty. *Comput. Linguist.*, 38(2):335–367.
- Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation, Marrakech, Morocco*.

# Mining Informal Language from Chinese Microtext: Joint Word Recognition and Segmentation

Aobo Wang<sup>1</sup>

Min-Yen Kan<sup>1,2\*</sup>

<sup>1</sup> Web IR / NLP Group (WING)

<sup>2</sup> Interactive and Digital Media Institute (IDMI)

National University of Singapore

13 Computing Link, Singapore 117590

{wangaobo, kanmy}@comp.nus.edu.sg

## Abstract

We address the problem of informal word recognition in Chinese microblogs. A key problem is the lack of word delimiters in Chinese. We exploit this reliance as an opportunity: recognizing the relation between informal word recognition and Chinese word segmentation, we propose to model the two tasks jointly. Our joint inference method significantly outperforms baseline systems that conduct the tasks individually or sequentially.

## 1 Introduction

User generated content (UGC) – including microblogs, comments, SMS, chat and instant messaging – collectively referred to as *microtext* by Gouwset *et al.* (2011) or *network informal language* by Xia *et al.* (2005), is the hallmark of the participatory Web.

While a rich source that many applications are interested in mining for knowledge, microtext processing is difficult to process. One key reason for this difficulty is the ubiquitous presence of informal words – anomalous terms that manifest as *ad hoc* abbreviations, neologisms, unconventional spellings and phonetic substitutions. Such informality is often present in oral conversation, and user-generated microblogs reflect this informality. Natural language processing (NLP) tools largely fail to work properly on microtext, as they have largely been trained on formally written text (i.e., newswire). Recent work has started to address these shortcomings (Xia and Wong, 2006; Kobus *et al.*, 2008; Han and Baldwin, 2011). Informal words and their usage in microtext evolves quickly, following social trends and news events.

\*This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

These characteristics make it difficult for lexicographers to compile lexica to keep with the pace of language change.

We focus on this problem in the Chinese language. Through our analysis of a gathered Chinese microblog corpus, we observe that Chinese informal words originate from three primary sources, as given in Table 1.

But unlike noisy words in English, Chinese informal words are more difficult to mechanically recognize for two critical reasons: first, Chinese does not employ word delimiters; second, Chinese informal words combine numbers, alphabetic letters and Chinese characters. Techniques for English informal word detection that rely on word boundaries and informal word orthography need to be adapted for Chinese. Consider the microtext “不要剧透了” (meaning “Don’t tell me the spoilers (to a movie or joke)”, also in Table 1). If “不要” (“don’t”) and “了” (past tense marker) are correctly recognized as two words, we may predict the previously unseen characters “剧透” (“tell spoilers”) as an informal word, based on the learned Chinese language patterns. However, state-of-the-art Chinese segmenters<sup>1</sup> incorrectly yield “不要\_剧\_透了”, preferring to chunk “透了” (“thoroughly”) as a word, as they do not consider the possibility that “剧透” (“spoiler”) could be an informal word. This example illustrates the mutual dependency between Chinese word segmentation (henceforth, CWS) and informal word recognition (IWR) that should be solved jointly.

Hence, rather than pipeline the two processes serially as previous work, we formulate it as a two-layer sequential labeling problem. We employ factorial conditional random field (FCRF) to solve both CWS and IWR jointly. To our best knowledge, this is the first work that shows how Chinese microtext can be analyzed from raw text to

<sup>1</sup><http://www.ictclas.org/index.html>

Table 1: Our classification of Chinese informal words as originating from three primary sources. For **Phonetic Substitutions**, pronunciation is indicated by the phonetic Pinyin transcription system.

	Informal Word	Formal Word	Example Sentence	English Translation
1) Phonetic Substitutions	木有(mu4 you3)	没有(me1 you3)	开发区木有出租车	No taxi in the development area
	孩纸们(hai2 zhi3 men)	孩子们(hai2 zi men)	起床了孩纸们	Get up kids
2) Abbreviation	bs	鄙视(bi shi)	我bs你	I despise you
	桌游	桌面游戏	来桌游吧	Let's play board games
3) Neologisms	剧透	剧情透露	不要剧透了	Don't tell (me) the spoilers
	给力	很棒	真给力啊	So awesome!
	秒杀	迅速购买	速度秒杀它	Quickly purchase it

derive joint solutions for both problems of CWS and IWR. We also propose novel features for input to the joint inference. Our techniques significantly outperform both research and commercial state-of-the-art for these problems, including two-step linear CRF baselines which perform the two tasks sequentially.

We detail our methods in Section 2. In Section 3, we first describe the details of our dataset and baseline systems, followed by demonstrating two sets of experiments for CWS and IWR, respectively. Section 4 offers the discussion on error analysis and limitations. We discuss related work in Section 5, before concluding our paper.

## 2 Methodology

Given an input Chinese microblog post, our method simultaneously segments the sentences into words (the Chinese Word Segmentation, CWS, task), and marks the component words as informal or formal ones (the Informal Word Recognition, IWR, task).

### 2.1 Problem Formalization

The two tasks are simple to formalize. The IWR task labels each Chinese character with either an **F** (part of a formal word) or **IF** (informal word). For the CWS task, we follow the widely-used **BIES** coding scheme (Low et al., 2005; Hai et al., 2006), where **B**, **I**, **E** and **S** stand for *beginning of a word*, *inside a word*, *end of a word* and *single-character word*, respectively. As a result, we have two (hidden) labels to associate with each (observable) character. Figure 1 illustrates an example microblog post graphically, where the labels are in circles and the observations are in squares. The two informal words in the example post are “木有” (normalized form: “没有”; English gloss: “no”) and “rp” (“人品值”; “luck”).

### 2.2 Conditional Random Field Models

Given the general performance and discriminative framework, Conditional Random Fields (CRFs) (Lafferty et al., 2001) is a suitable framework for tackling sequence labeling problems. Other alternative frameworks such as Markov Logic Networks (MLNs) and Integer Linear Programming (ILP) could also be considered. However, we feel that for this task, formulating efficient global formulas (constraints) for MLN (ILP) is comparatively less straightforward than in other tasks (e.g. compared to Semantic Role Labeling, where the rules may come directly from grammatical constraints). CRFs represent a basic, simple and well-understood framework for sequence labeling, making it a suitable framework for adapting to perform joint inference.

#### 2.2.1 Linear-Chain CRF

A linear-chain CRF (LCRF; Figure 2a) predicts the output label based on feature functions provided by the scientist on the input. In fact, the LCRF has been used for the exact problem of CWS (Sun and Xu, 2011), garnering state-of-the-art performance, and as such, validate it as a strong baseline for comparison.

#### 2.2.2 Factorial CRF

To properly model the interplay between the two sub-problems, we employ the factorial CRF (FCRF) model, which is based on the dynamic CRF (DCRF) (Sutton et al., 2007). By introducing a pairwise factor between different variables at each position, the FCRF model results as a special case of the DCRF. A FCRF captures the joint distribution among various layers and jointly predicts across layers. Figure 2 illustrates both the LCRF and FCRF models, where cliques include within-chain edges (e.g.,  $y_t, y_{t+1}$ ) in both LCRF and FCRF models, and the between-chain edges (e.g.,  $y_t, z_t$ ) only in the FCRF.

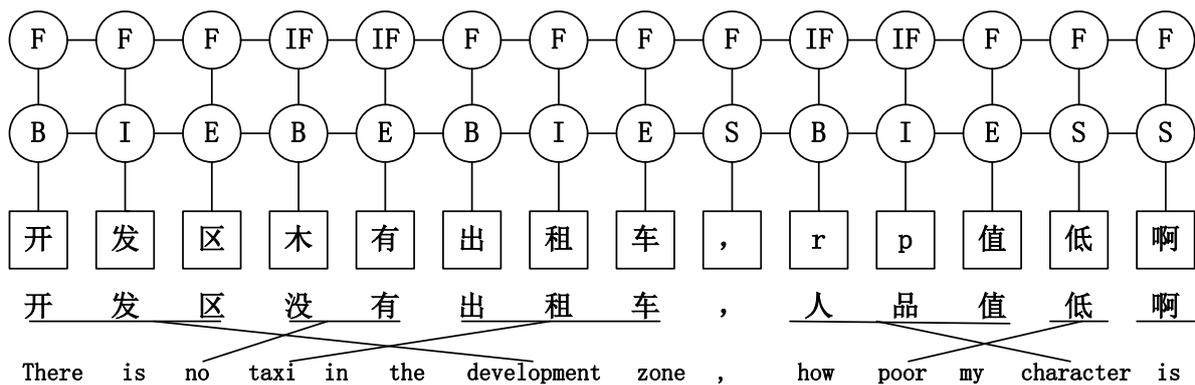


Figure 1: A Chinese microtext (bottom layer) with annotations for IWR (top layer) and CWS (middle layer). The bottom three lines give the normalized Chinese form, its pronunciation in Pinyin and aligned English translation.

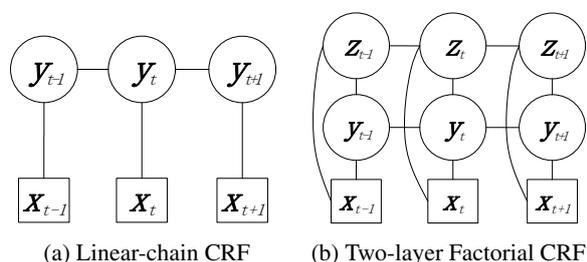


Figure 2: Graphical representations of the two types of CRFs used in this work.  $y_t$  denotes the 1<sup>st</sup> layer label,  $z_t$  denotes the 2<sup>nd</sup> layer label, and  $x_t$  denotes the observation sequence.

Although the FCRF can be collapsed into a LCRF whose state space is the cross-product of the outcomes of the state variables (i.e., 8 labels in this case), Sutton *et al.* (2007) noted that such a LCRF requires not only more parameters in the number of variables, but also more training data to achieve equivalent performance with an FCRF. Given the limited scale of the state space and training data, we follow the FCRF model, using exact Junction Tree (Jensen, 1996) inference and decoding algorithm to perform prediction.

### 2.3 CRF Features

We use three broad feature classes – lexical, dictionary-based and statistical features – aiming to distinguish the output classes for the CWS and IRW problems. Character-based sequence labeling is employed for word segmentation due to its simplicity and robustness to the unknown word problem (Xue, 2003).

A key contribution of our work is also to propose novel features for joint inference. We

propose new features for the dictionary-based and statistical feature classes, which we have marked in the discussion below with “(\*)”. We later examine their efficacy in Section 3.

**Lexical Features.** As a foundation, we employ lexical (n-gram) features informed by the previous state-of-the-art for CWS (Sun and Xu, 2011; Low *et al.*, 2005). These features are listed below<sup>2</sup>:

- Character 1-gram:  $C_k(i - 4 < k < i + 4)$
- Character 2-gram:  $C_k C_{k+1}(i - 4 < k < i + 3)$
- Character 3-gram:  $C_k C_{k+1} C_{k+2}(i - 3 < k < i + 2)$
- Character lexicon:  $C_{-1} C_1$   
This feature is used to capture the common indicators in Chinese interrogative sentences. (e.g., “是不是” (“whether or not”), “好不好” (“OK or not”))
- Whether  $C_k$  and  $C_{k+1}$  are identical, for  $i - 4 < k < i + 3$ .  
This feature is used to capture the words of employing character doubling in Chinese. (e.g., “拜拜” (“see you”), “天天” (“every day”))

**Dictionary-based Features.** We use features that indicate whether the input character sequence

<sup>2</sup>For notational convenience, we denote a candidate character token  $C_i$  as having a context  $\dots C_{i-1} C_i C_{i+1} \dots$ . We use  $C_{m:n}$  to express a subsequence starting at the position  $m$  and ending at  $n$ .  $len$  stands for the length of the subsequence, and  $offset$  denotes the position offset of  $C_{m:n}$  from the current character  $C_i$ . We use  $b$  (beginning),  $m$  (middle) and  $e$  (ending) to indicate the position of  $C_k$  ( $m \leq k \leq n$ ) within the string  $C_{m:n}$ .

matches entries in certain lexica. We use the on-line dictionary from Peking University as the formal lexicon and the compiled informal word list from our training instances as the informal lexicon. In addition, we employ additional online word lists<sup>3</sup> to distinguish named entities and function words from potential informal words.

As shown in Table 1, alphabetic sequences in microblogs may refer to Chinese Pinyin or Pinyin abbreviations, rather than English (e.g., “*bs*” for *bi shi*; “to despise”). Hence, we added dictionary-based features to indicate the presence of Pinyin initials, finals and standard Pinyin expansions, using a UK English word list<sup>4</sup>. The final list of dictionary-based features employed are:

- If  $C_k (i - 4 < k < i + 4)$  is a surname:  
*Surname@k*
- (\*) If  $C_k (i - 4 < k < i + 4)$  is a stop word:  
*StopW@k*
- (\*) If  $C_k (i - 4 < k < i + 4)$  is a noun-suffix:  
*NSuffix@k*
- (\*) If  $C_k (i - 4 < k < i + 4)$  is a Pinyin Initial: *Initial@k*
- (\*) If  $C_k (i - 4 < k < i + 4)$  is a Pinyin Final: *Final@k*
- If  $C_k (i - 4 < k < i + 4)$  is a English letter:  
*En@k*
- If  $C_{m:n} (i - 4 < m < n < i + 4, 0 < n - m < 5)$  matches one entry in the Peking University dictionary:  
*FW@m:n; len@offset; FW-C\_k@b-offset, FW-C\_k@n-offset or FW-C\_k@e-offset*
- (\*) If  $C_{m:n} (i - 4 < m < n < i + 4, 0 < n - m < 5)$  matches one entry in the informal word list:  
*IFW@m:n; len@offset; IFW-C\_k@b-offset, IFW-C\_k@n-offset or IFW-C\_k@e-offset*
- (\*) If  $C_{m:n} (i - 4 < m < n < i + 4, 0 < n - m < 5)$  matches one entry in the valid Pinyin list:  
*PY@m:n; len@offset; PY-C\_k@b-offset, PY-C\_k@n-offset or PY-C\_k@e-offset*

**Statistical Features.** We use pointwise mutual information (PMI) variant (Church and Hanks,

<sup>3</sup>Resources are available at <http://www.sogou.com/labs/resources.html>

<sup>4</sup><http://www.bckelk.uklinux.net/menu.html>

1990) to account for global, corpus-wide information. This measures the difference between the observed probability of an event (i.e., several characters combined as an informal word) and its expectation, based on the probabilities of the individual events (i.e., the probability of the individual characters occurring in the corpus). Compared with other standard association measures such as MI, PMI tends to assign rare events higher scores. This makes it a useful signal for IWR, as it is sensitive to informal words which often have low frequency. However, the word frequency alone is not reliable enough to distinguish informal words from uncommon but formal words.

In response to these difficulties in differentiating linguistic registers, we compute two different PMI scores for character-based bigrams from two large corpora representing news and microblogs as features. We also use the difference between the two PMI scores as a differential feature. In addition, we also convert all the character-based bigrams into Pinyin-based bigrams (ignoring tones<sup>5</sup>) and compute the Pinyin-level PMI in the same way. These features capture inconsistent use of the bigram across the two domains, which assists to distinguish informal words. Note that we eschew smoothing in our computation of PMI, as it is important to capture the inconsistent character bigrams usage between the two domains. For example, the word “*rp*” appears in the microblog domain, but not in news. If smoothing is conducted, the character bigram “*rp*” will be given a non-zero probability in both domains, not reflective of actual use. For each character  $C_i$ , we incorporate the PMI of the character bigrams as follows:

- (\*) If  $C_k C_{k+1} (i - 4 < k < i + 4)$  is not a Chinese word recorded in dictionaries:  
*CPMI-N@k+i; CPMI-M@k+i; CDiff@k+i; PYPMI-N@k+i; PYPMI-M@k+i; PYDiff@k+i*

### 3 Experiment

We discuss the dataset, baseline systems and experiments results in detail in the following.

#### 3.1 Data Preparation

We utilize the Chinese social media archive, PrEV (Cui et al., 2012), to obtain Chinese mi-

<sup>5</sup>The informal word may have the same Pinyin transcription as its formal counterpart without considering the differences in tones.

croblog posts from the public timeline of Sina Weibo<sup>6</sup>. Sina Weibo is the largest microblogging in China, where over 100 million Chinese microblog posts are posted daily (Cao, 2012), likely the largest public source of informal and daily Chinese language use. Our dataset has a total of 6,678,021 messages, covering two months from June to July of 2011. To annotate the corpus, we employ *Zhubajie*<sup>7</sup>, one of China mainland’s largest crowdsourcing (Wang et al., 2010) platforms to obtain informal word annotations. In total, we spent US\$110 on assembling a subset of 5,500 posts (12,446 sentences) in which 1,658 unique informal words are annotated within five weeks via *Zhubajie*. Each post was annotated by three annotators with moderate (0.57) inter-annotator agreement measured by Fleiss’  $\kappa$  (Joseph, 1971), and conflicts were resolved by majority voting.

We divided the annotated corpus, taking 4,000 posts for training, and the remainder (1,500) for testing. Through inspection, we note that 79.8% of the informal words annotated in the testing set are not covered by the training set. We also follow Wang et al. (2012)’s conventions and apply rule-sets to preprocess the corpus’ *URLs*, *emoticons*, “*@usernames*” and *Hashtags* as pre-segmented words, before input to CWS and IWR. For the CWS task, the first author manually labelled the same corpus following the segmentation guidelines published with the *SIGHAN-5*<sup>8</sup> MSR dataset.

### 3.2 Baseline Systems

We implemented several baseline systems to compare with proposed FCRF joint inference method.

**Existing Systems.** We re-implemented Xia and Wong (2008)’s extended Support Vector Machine (SVM) based microtext IWR system to compare with our method. Their system only does IWR, using the CWS and POS tagging output of the ICTCLAS segmenter (Zhang et al., 2003) as input. To compare our joint inference versus other learning models, we also employed a decision tree (DT) learner, equipped with the same feature set as our FCRF. Both the SVM and DT models are provided by the Weka3 (Hall et al., 2009) toolkit, using its default configuration.

To evaluate CWS performance, we compare with two recent segmenters. Sun and Xu (2011)’s

work achieves state-of-the-art performance and is publicly available. They employ a LCRF taking as input both lexical and statistical features derived from unlabeled data. As a second baseline, we also evaluate against a widely-used, commercially-available alternative, the recently released 2011 ICTCLAS segmenter<sup>9</sup>.

**Two-stage Sequential Systems.** To benchmark the improvement that the factorial CRF model has by doing the two tasks jointly, we compare with a LCRF solution that chains these two tasks together. For completeness, we test pipelining in both directions – CWS feeding features for IWR ( $LCRF_{cws} \succ LCRF_{iwr}$ ), and the reverse ( $LCRF_{iwr} \succ LCRF_{cws}$ ). We modify the open-source Mallet GRMM package (Sutton, 2006) to implement both this sequential LCRF model and our proposed FCRF model. Both models take the whole feature set described in Section 2.3.

**Upper Bound Systems.** To measure the upper-bound achievable with perfect support from the complementary task, we also provided gold standard labels of one task (e.g., IWR) as an input feature to the other task (e.g., CWS). These systems (hereafter denoted as LCRF $\succ$ LCRF-UB and FCRF-UB) are meant for reference only, as they have access to answers for the opposing tasks.

**Adapted SVM for Joint Classification.** For completeness, we also compared our work against the standard SVM classification model that performs both tasks by predicting the cross-product of the CWS and IWR individual classes (in total, 8 classes). We train the SVM classifier on the same set of features as the FCRF, by providing the cross-product of two layer labels as gold labels. This system (hereafter denoted as SVM-JC) was implemented using the LibSVM package (Chang and Lin, 2011).

### 3.3 Evaluation Metrics

We use the standard metrics of precision, recall and  $F_1$  for the IWR task. Only words that exactly match the manually-annotated labels are considered correct. For example given the sentence “怎么介么好吃呢” (“怎么这么好吃呢”; “How delicious it is”), if the IWR component identifies “介么” as an informal word, it will be considered correct, whereas both “介么好” and “介” are deemed incorrect. For CWS evaluation, we employ the conventional scoring script provided in *SIGHAN-*

<sup>6</sup><http://open.weibo.com>

<sup>7</sup><http://www.zhubajie.com>

<sup>8</sup><http://www.sighan.org>

<sup>9</sup><http://www.ictclas.org/index.html>

5, which also provides out-of-vocabulary recall (OOVR).

To determine statistical significance of the improvements, we also compute paired, one-tailed  $t$  tests. As pointed out by Yeh and Alexander (2000), the randomization method is more reliable in measuring the significance of  $F_1$  through handling non-linear functions of random variables. Thus we employ Padó (2006)’s implementation of randomization algorithm to measure the significance of  $F_1$ .

### 3.4 Experimental Results

The goal of our experiments is to answer the following research questions:

- RQ1 Do the two tasks of CWS and IWR benefit from each other?
- RQ2 Is jointly modeling both tasks more efficient than conducting each task separately or sequentially?
- RQ3 What is the upper bound improvement that can be achieved with perfect support from the opposing task?
- RQ4 Are the features we designed for the joint inference method effective?
- RQ5 Is there a significant difference between the performance of the joint inference of a cross-product SVM and our proposed FCRF?

#### 3.4.1 CWS Performance

Table 2: Performance comparison on the CWS task. The two bottom-most rows show upper bound performance. ‘ $\ddagger$ ’(‘\*’) in the top four lines indicates statistical significance at  $p < 0.001$  (0.05) when compared with the previous row. Symbols in the bottom two lines indicate significant difference between upper bound systems and their corresponding counterparts.

	Pre	Rec	$F_1$	OOVR
ICTCLAS (2003)	0.640	0.767	0.698	0.551
Sun and Xu (2011)	0.661 $\ddagger$	0.691 $\ddagger$	0.675	0.572 $\ddagger$
LCRF <sub>iwr</sub> > LCRF <sub>cws</sub>	0.741 $\ddagger$	0.775 $\ddagger$	0.758*	0.607*
FCRF	<b>0.757<math>\ddagger</math></b>	<b>0.801<math>\ddagger</math></b>	<b>0.778*</b>	<b>0.633*</b>
LCRF <sub>iwr</sub> > LCRF <sub>cws</sub> -UB	0.807 $\ddagger$	0.815 $\ddagger$	0.811*	0.731 $\ddagger$
FCRF-UB	0.820 $\ddagger$	0.833 $\ddagger$	0.826*	0.758 $\ddagger$

In general, our FCRF yields the best performance among all systems (top portion of Table 2),

answering RQ1. Given microblog posts as test data, the  $F_1$  of ICTCLAS drops from 0.985<sup>10</sup> to 0.698, clearly showing the difficulty of processing microtext. The sequential LCRF model and FCRF model both outperform the baselines, which means with the novel features shared by the two tasks, CWS benefits significantly from the results of IWR. Hence our segmenter outperforms the existing segmenters by tackling one of the bottlenecks of recognizing informal words in Chinese microtext.

To illustrate, the sequence “...有木有人...” (“...有没有人...”; “...is there anyone...”), is correctly labeled as **BIES** by our FCRF model but mislabeled by baseline systems as **SSBE**. This is likely due to the ignorance of the informal word “有木有”, leading baseline systems to keep the formal word “有人” (“someone”) as a segment.

More importantly, by jointly optimizing the probabilities of labels on both layers, the FCRF model slightly but significantly improves over the sequential LCRF method, answering RQ2. Thus we conclude that jointly modeling both tasks is more effective than performing the tasks sequentially.

For RQ3, the last two rows presents the upper-bound systems that have access to gold standard labels for IWR. Both upper-bound systems statistically outperform their counterparts, indicating that there is still room to improve CWS performance with better IWR as input. This also validates our assumption that CWS can benefit from joint consideration of IWR. Taking the best previous work as our lower bound (0.69  $F_1$ ), we see that our FCRF methodology (0.77) makes significant progress towards the upper bound (0.82).

#### 3.4.2 IWR Performance

For RQ1 and RQ2, Table 3 compares the performance of our method with the baseline systems on the IWR task. Overall, the FCRF method again outperforms all the baseline systems. We note that the CRF based models achieve much higher precision score than baseline systems, which means that the CRF based models can make accurate predictions without enlarging the scope of prospective informal words. Compared with the CRF based models, the SVM and DT both over-predict informal words, incurring a larger precision penalty. Studying this phe-

<sup>10</sup>Self-declared segmentation accuracy on formal text. <http://www.ictclas.org/>

Table 3: Performance comparison on the IWR task. ‘‡’ or ‘\*’ in the top four rows indicates statistical significance at  $p < 0.001$  or  $< 0.05$  compared with the previous row. Symbols in the bottom two rows indicate differences between upper bound systems and their counterparts.

	Pre	Rec	F <sub>1</sub>
<b>SVM</b>	0.382	0.621	0.473
<b>DT</b>	0.402*	0.714*	0.514*
<b>LCRF<sub>cws</sub> &gt; LCRF<sub>iwr</sub></b>	0.858‡	0.591‡	0.699*
<b>FCRF</b>	<b>0.877*</b>	<b>0.655*</b>	<b>0.750*</b>
<b>LCRF<sub>cws</sub> &gt; LCRF<sub>iwr</sub>-UB</b>	0.840	0.726*	0.779*
<b>FCRF-UB</b>	0.878	0.752*	0.810*

nomenon more closely, we find it is difficult for the baseline systems to classify segments mixed with formal and informal characters. Taking the microblog “怎么介么好吃呢” (“怎么这么好吃呢”; “how delicious it is”) as an example, without considering the possible word boundaries suggested by the contextual formal words – i.e., “怎么” (“how”) and “好吃” (“delicious”) – the baselines chunk the informal words (i.e., “介么”) together with adjacent characters mistakenly as “介么好” or “么介么”.

As indicated by the bold figures in Table 3, the FCRF performs slightly better than the sequential LCRF ( $p < 0.05$ ) – a weaker trend when compared with the CWS case. As an example, the sequential LCRF method fails to recognize “爱疯” (“iPhone”) as an informal word in the sentence “我的爱疯好玩” (“my iPhone is fun”), where the FCRF succeeds. Inspecting the output, the LCRF segmenter mislabels “爱疯” as **SS**. By jointly considering the probabilities of the two layers, the FCRF model infers better quality segmentation labels, which in turn enhances the FCRF’s capability to recognize the sequence of two characters as an informal word. This is further validated by the significant performance gulf between the upper bound and the basic system shown in the lower half of the table.

For RQ3, interestingly, the difference in performance between the LCRF and FCRF upper-bound systems is not significant. However, these are upper bounds, and we expect on real-world data that CWS performance will not be perfect. As such, we still recommend using the FCRF model, as the joint process is more robust to noisy input from one channel.

Table 4: F<sub>1</sub> comparison between FCRF and FCRF<sub>-new</sub>. (“\*”) indicates statistical significance at  $p < 0.05$  when compared with the previous row.

	CWS	IWR
<b>FCRF<sub>-new</sub></b>	0.690	0.552
<b>FCRF</b>	0.778*	0.750*

### 3.4.3 Feature set evaluation

For RQ4, to evaluate the effectiveness of our newly-introduced feature sets (those marked with “\*” in Section 2.3), we also test a FCRF (FCRF<sub>-new</sub>) without our new features. According to Table 4, performance drops by a significant amount: 0.088 F<sub>1</sub> on CWS and 0.198 F<sub>1</sub> on IWR. FCRF<sub>-new</sub> makes many mistakes identical to the baselines: segmenting informal words into several single-character words and chunking adjacent characters from informal and formal words together.

### 3.4.4 Adapted SVM-JC vs. FCRF

Table 5: F<sub>1</sub> comparison between SVM, SVM-JC and FCRF. ‘‡’ (“\*”) indicates statistical significance at  $p < 0.001$  (0.05) when compared with the previous row.

	CWS	IWR
<b>SVM</b>	—	0.473
<b>SVM-JC</b>	0.741	0.624‡
<b>FCRF</b>	0.778*	0.750*

For RQ5, according to Table 5, our SVM trained to predict the cross-product CWS/IWR classification (SVM-JC) performs quite well on its own. Unsurprisingly, it does not outperform our proposed FCRF, which has access to more structural correlation among the CWS and IWR labels. SVM-JC significantly ( $p < 0.001$ ) outperforms the baseline SVM system by 0.151 in the IWR task, which we think is partially explained by its good performance (0.761) on the CWS task. The over-prediction tendency of the individual SVM is largely solved by simultaneously modeling the CWS task, whereas FCRF turns out to be more effective in solving joint inference problem, although in a weaker trend in terms of the statistical significance ( $p < 0.05$ ).

We conclude that the use of the FCRF model and the addition of our new features are both essential for the high performance of our system.

## 4 Discussion

We wish to understand the causes of errors in our models so that we may better understand its weaknesses. Manually inspecting the errors of our system, we found three major categories of errors which we dissect here.

For IWR, the major source of error, accounting for more than 60% of all errors, is caused by what we term the *partially observed informal word* phenomenon. This refers to informal words containing multiple characters, where some of its components have appeared in the training data as informal words individually. For instance, the single-character informal word, “狠” (“很”; “very”) appears in training multiple times, thus the unseen informal word “狠久” (“很久”; “long time”) is a *partially observed* informal word. In this case, the model incorrectly labels the known, single character “狠” with **IF.S** as an informal word, instead of labeling the unseen sequence “狠久” with correct labels **IF.B IF.E**. Errors then result in both tasks.

This observation motivates the use of the relation between the known informal word and its formal counterpart in order to inform the model to better predict in cases of partial observations. Following the same example, given that “狠” is an informal word, if the model also considers the probability of normalizing “狠” to “很”, while considering the higher probability that the character sequence “很久” could be a formal word, there would be a higher likelihood of correctly predicting the sequence “狠久” as an informal word. So informal word normalization is also an intrinsic component of IWR and CWS, and we believe it is an interesting direction for future work.

Another source of error is a side effect of microtext being extremely short. For example, in the sentence “肥家! 太累了。。。 ” (“回家! 太累了。。。”; “Go home! Exhausted.”), the unseen informal word “肥家” itself forms a short sentence. Although it has a subsequent sentence “太累了。。。 ” (“Exhausted”) as context, and the two are pragmatically related, (i.e., “I am exhausted! [And as a result,] I want to go home.”), the lexical relationship between the sentences is weak; i.e., “太累了。。。 ” appears frequently as the context of various sentences, making the context difficult to utilize. These phenomena makes it difficult to recognize “肥家” as an informal word.

A possible solution could factor in proximity, similar to density-based matching, as in Tellex *et*

Table 6: Sample Chinese freestyle named entities that are usernames.

Freestyle Named Entity	Explanation
“榴莲雪媚娘”	“榴莲” (“durian”), “雪” (“snow”), “媚娘” (“charming lady”)
“棉宝”	It is short for the cartoon name “海绵宝宝”.
“dj文祥”, “徐pp”	Usernames mixed of Chinese and alphabetic characters

*al.* (2003). We can assign a higher weight to features related to characters closer to the current target character. In particular, for this example, given the current target character “肥”, we can assign higher weight to features generated from features from the proximal context “肥家”, and lower weight to features extracted from distal contexts.

Another major group of errors come from what we term *freestyle named entities* as exemplified in Table 6; i.e., person names in the form of user IDs and nicknames, that have less constraint on form in terms of length, canonical structure (not surnames with given names; as is standard in Chinese names) and may mix alphabetic characters. Most of these belong to the category of *Person Name* (PER), as defined in CoNLL-2003<sup>11</sup> Named Entity Recognition shared task. Such freestyle entities are often misrecognized as informal words, as they share some of the same stylistic markings, and are not marked by features used to recognize previous Chinese named entity recognition methods (Gao et al., 2005; Zhao and Kit, 2008) that work on news or general domain text. We recognize this as a challenge in Chinese microtext, but beyond the scope of our current work.

## 5 Related Work

In English, IWR has typically been investigated alongside normalization. Several recent works (Han and Baldwin, 2011; Gouws et al., 2011; Han et al., 2012) aim to produce informal/formal word lexicons and mappings. These works are based on distributional similarity and string similarity that address concerns of lexical variation and spelling. These methods propose two-step unsupervised approaches to first detect and then normalize detected informal words using dictionaries.

In processing Chinese informal language, work conducted by Xia and Wong address the problem

<sup>11</sup><http://www.cnts.ua.ac.be/conll2003/ner/>

of in bulletin board system (BBS) chats. They employ pattern matching and SVM-based classification to recognize Chinese informal sentences (not individual words) chat (Xia et al., 2005). Both methods had their advantages: the learning-based method did better on recall, while the pattern matching performed better on precision. To obtain consistent performance on new unseen data, they further employed an error-driven method which performed more consistently over time-varying data (Xia and Wong, 2006). In contrast, our work identifies individual informal words, a finer-grained (and more difficult) task.

While seminal, we feel that the difference in scope (informal sentence detection rather than word detection) shows the limitation of their work for microblog IWR. Their chats cover only 651 unique informal words, as opposed to our study covering almost triple the word types (1,658). Our corpus demonstrates a higher ratio of informal word use (a new informal word appears in  $\frac{1,658}{12,446} = 13\%$  of sentences, as opposed to  $\frac{651}{22,400} = 2\%$  in their BBS corpus). Further analysis of their corpus reveals that phonetic substitution is the primary origin of informal words in their corpus – 99.2% as reported in (Wong and Xia, 2008). In contrast, the origin for informal words in microblogs is more varied, where phonetic substitutions abbreviations and neologisms, account for 53.1%, 21.4% and 18.7% of the informal word types, respectively. Their method is best suited for phonetic substitution, thus performing well on their corpus but poorly on ours.

More closely related, Li and Yarowsky (2008) tackle Chinese IWR. They bootstrap 500 informal/formal word pairs by using manually-tuned queries to find definition sentences on the Web. The resulting noisy list is further re-ranked based on n-gram co-occurrence. However, their method makes a basic assumption that informal/formal word pairs co-occur within a definition sentence (i.e., “<informal word> means <formal word>”) may not hold in microblog data, as microbloggers largely do not define the words they use.

Closely related to our work is the task of Chinese new word detection, normally treated as a separate process from word segmentation in most previous works (Chen and Bai, 1998; Wu and Jiang, 2000; Chen and Ma, 2002; Gao et al., 2005). Aiming to improve both tasks, work by Peng *et al.* (2004) and Sun *et al.* (2012) conduct

segmentation and detection sequentially, but in an iterative manner rather than joint. This is a weakness as their linear CRF model requires re-training. Their method also requires thresholds to be set through heuristic tuning, as to whether the segmented words are indeed new words. We note that the task of new word detection refers to out-of-vocabulary (OOV) detection, and is distinctly different from IWR (new words could be both formal or informal words).

## 6 Conclusion

There is a close dependency between Chinese word segmentation (CWS) and informal word recognition (IWR). To leverage this, we employ a factorial conditional random field to perform both tasks of CWS and IWR jointly.

We propose novel features including statistical and lexical features that improve the performance of the inference process. We evaluate our method on a manually-constructed data set and compare it with multiple research and industrial baselines that perform CWS and IWR individually or sequentially. Our experimental results show our joint inference model yields significantly better  $F_1$  for both tasks. For analysis, we also construct upper bound systems to assess the potential maximal improvement, by feeding one task with the gold standard labels from the complementary task. These experiments further verify the necessity and effectiveness of modeling the two tasks jointly, and point to the possibility of even better performance with improved per-task performance.

Analyzing the classes of errors made by our system, we identify a promising future work topic to handle errors arising from *partially observed informal words* – where parts of a multi-character informal word have been observed before. We believe incorporating informal word normalization into the inference process may help address this important source of error.

## Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. We also appreciate the proofreading effort made by Tao Chen, Xiangan He, Ning Fang, Yushi Wang and Haochen Zhan from WING. This work also benefits from the discussion with Yang Liu, associate professor from Tsinghua University.

## References

- Belinda Cao. 2012. Sina's weibo outlook buoys internet stock gains: China overnight.
- Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, pages 27:1–27:27.
- Keh-Jiann Chen and Ming-Hong Bai. 1998. Unknown Word Detection for Chinese by a Corpus-Based Learning Method. *International Journal of Computational Linguistics and Chinese Language Processing*, pages 27–44.
- Keh-Jiann Chen and Wei-Yun Ma. 2002. Unknown Word Extraction for Chinese Documents. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7.
- Kenneth Ward Church and Patrick Hanks. 1990. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistic*, pages 22–29.
- Anqi Cui, Liner Yang, Dejun Hou, Min-Yen Kan, Yiqun Liu, Min Zhang, and Shaoping Ma. 2012. PrEV: Preservation Explorer and Vault for Web 2.0 User-Generated Content. *Theory and Practice of Digital Libraries*, pages 101–112.
- Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. 2005. Chinese Word Segmentation and Named Entity Recognition: A Pragmatic Approach. *Computational Linguistic*, pages 531–574.
- Stephan Gouws, Donald Metzler, Congxing Cai, and Eduard Hovy. 2011. Contextual Bearing on Linguistic Variation in Social Media. In *Proceedings of the Workshop on Language in Social Media*, pages 20–29.
- Zhao Hai, Huang Chang-Ning, Li Mu, and Lu Bao-Liang. 2006. Effective Tag Set Selection in Chinese Word Segmentation via Conditional Random Field Modeling. *The 20th Pacific Asia Conference on Language, Information and Computation*, pages pp.87–94.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *ACM Special Interest Groups on Knowledge Discovery and Data Mining Explorations Newsletter*, pages 10–18.
- Bo Han and Timothy Baldwin. 2011. Lexical Normalisation of Short Text Messages: Makn Sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–378.
- Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically Constructing a Normalisation Dictionary for Microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432.
- Finn V Jensen. 1996. *An Introduction to Bayesian Networks*, volume 74.
- Fleiss L Joseph. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, pages 378–382.
- Catherine Kobus, François Yvon, and Géraldine Damnati. 2008. Normalizing SMS: Are Two Metaphors Better Than One? In *International Conference on Computational Linguistics*, pages 441–448.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.
- Zhifei Li and David Yarowsky. 2008. Mining and Modeling Relations between Formal and Informal Chinese Phrases from Web Corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1031–1040.
- Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. 2005. A Maximum Entropy Approach to Chinese Word Segmentation. *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- Sebastian Padó, 2006. *User's Guide to SIGF: Significance Testing by Approximate Randomisation*.
- Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese Segmentation and New Word Detection Using Conditional Random Fields. In *Proceedings of the 20th international conference on Computational Linguistics*, page 562.
- Weiwei Sun and Jia Xu. 2011. Enhancing Chinese Word Segmentation Using Unlabeled Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 970–979.
- Xu Sun, Houfeng Wang, and Wenjie Li. 2012. Fast Online Training with Frequency-Adaptive Learning Rates for Chinese Word Segmentation and New Word Detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 253–262.
- Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. 2007. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. *Journal of Machine Learning Research*, pages 693–723.
- Charles Sutton. 2006. GRMM: GRaphical Models in Mallet. In URL <http://mallet.cs.umass.edu/grmm>.

- Stephanie Tellex, Boris Katz, Jimmy Lin, Aaron Fernandes, and Gregory Marton. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 41–47.
- Aobo Wang, Cong Duy Vu Hoang, and Min-Yen Kan. 2010. Perspectives on Crowdsourcing Annotations for Natural Language Processing, journal = Language Resources and Evaluation. pages 1–23.
- Aobo Wang, Tao Chen, and Min-Yen Kan. 2012. Retweeting From A Linguistic Perspective. In *Proceedings of the Second Workshop on Language in Social Media*, pages 46–55.
- Kam-Fai Wong and Yunqing Xia. 2008. Normalization of Chinese Chat Language. *Language Resources and Evaluation*, pages 219–242.
- Andi Wu and Zixin Jiang. 2000. Statistically-Enhanced New Word Identification in A Rule-based Chinese Aystem. In *Proceedings of the second workshop on Chinese Language Processing*, pages 46–51.
- Yunqing Xia and Kam-Fai Wong. 2006. Anomaly Detecting within Dynamic Chinese Chat Text. *NEW TEXT Wikis and blogs and other dynamic text sources*, page 48.
- Yunqing Xia, Kam-Fai Wong, and Wei Gao. 2005. NIL Is Not Nothing: Recognition of Chinese Network Informal Language Expressions. In *4th SIGHAN Workshop on Chinese Language Processing*, volume 5.
- Nianwen Xue. 2003. Chinese Word Segmentation as Character Tagging. *Computational Linguistics and Chinese Language Processing*, pages 29–48.
- Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th conference on Computational linguistics - Volume 2*, pages 947–953.
- Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. HHMM-based Chinese Lexical Analyzer ICTCLAS. In *Proceedings of the second SIGHAN workshop on Chinese language processing*, pages 184–187.
- Hai Zhao and Chunyu Kit. 2008. Unsupervised Segmentation Helps Supervised Learning of Character Tagging for Word Segmentation and Named Entity Recognition. In *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing*, pages 106–111.

# Generating Synthetic Comparable Questions for News Articles

**Oleg Rokhlenko**

Yahoo! Research

Haifa 31905, Israel

olegro@yahoo-inc.com

**Idan Szpektor**

Yahoo! Research

Haifa 31905, Israel

idan@yahoo-inc.com

## Abstract

We introduce the novel task of automatically generating questions that are relevant to a text but do not appear in it. One motivating example of its application is for increasing user engagement around news articles by suggesting relevant comparable questions, such as “*is Beyonce a better singer than Madonna?*”, for the user to answer. We present the first algorithm for the task, which consists of: (a) offline construction of a comparable question template database; (b) ranking of relevant templates to a given article; and (c) instantiation of templates only with entities in the article whose comparison under the template’s relation makes sense. We tested the suggestions generated by our algorithm via a Mechanical Turk experiment, which showed a significant improvement over the strongest baseline of more than 45% in all metrics.

## 1 Introduction

For companies whose revenues are mainly ad-based, *e.g.* Facebook, Google and Yahoo, increasing user engagement is an important goal, leading to more time spent on site and consequently to increased exposure to ads. Examples for typical engaging content include other articles for the user to read, updates from the user’s social neighborhood and votes or comments on videos, blogs etc.

In this paper we propose a new way to increase user engagement around news articles, namely suggesting questions for the user to answer, which are related to the viewed article. Our motivation is that there are questions that are “irresistible” because they are fun, involve emotional reaction and expect simple answers. These are comparative questions, such as “*is Beyonce a better singer than*

*Madonna?*”, “*who is better looking, Brad Pitt or George Clooney?*”, “*who is faster: Superman or Flash?*” and “*which camera brand do you prefer: Canon or Nikon?*” Furthermore, such questions are social in nature since users would be interested in reading the opinions of other users, similar to viewing other comments (Schuth et al., 2007). Hence, a user that provided an answer may return to view other answers, further increasing her engagement with the site.

One approach for generating comparable questions would be to employ traditional question generation, which syntactically transform assertions in a given text into questions (Mitkov et al., 2006; Heilman and Smith, 2010; Rus et al., 2010). Sadly, fun and engaging comparative questions are typically not found within the text of news articles. A different approach would be to find concrete relevant questions within external collections of manually generated comparable questions. Such collections include Community-based Question Answering (CQA) sites such as Yahoo! Answers and Baidu Zhidao and sites that are specialized in polls, such as Toluna. However, it is highly unlikely that such sources will contain enough relevant questions for any news article due to typical sparseness issues as well as differences in interests between askers in CQA sites and news reporters. To better address the motivating application above, we propose the novel task of automatically suggesting comparative questions that are relevant to a given input news article but do not appear in it.

To achieve broad coverage for our task, we present an algorithm that generates synthetic concrete questions from *question templates*, such as “*Who is a better actor: #1 or #2?*”. Our algorithm consists of two parts. An offline part constructs a database of comparative question templates that appear in a large question corpus. For a given news article, an online part chooses relevant tem-

## Who's Hollywood's Hottest Dad?

In honor of Father's Day, Us Weekly wants to know -- who do you think is the hottest dad in Hollywood?

**David Beckham**, 36 -- who's expecting baby No. 4 with wife **Victoria** -- is already father to sons Brooklyn, 12, Romeo, 8, and Cruz, 6; **Brad Pitt**, 47, co-parents six children with **Angelina Jolie** (Maddox, 9, Pax, 7, Zahara, 6, Shiloh, 5, and twins Knox and Vivienne, 2); **Will Smith** helped pave the way for his superstar children Jaden, 12, and Willow, 10; and **Matthew McConaughey**, 41, is the proud father of cuties Levi, 2, and Vida, 1.

Tell Us: Who's the hottest dad in Hollywood?

Figure 1: An example news article from OMG!

plates for the article by matching between the article content and typical template contexts. The algorithm then instantiates each relevant template with two entities that appear in the article. Yet, for a given template, only some of the entities are plausible slot fillers. For example, ‘*Madonna*’ is not a reasonable filler for “*Who is a better dad, #1 or #2?*”. Thus, our algorithm employs entity filtering to exclude candidate instantiations that do not make sense.

To test the performance of our algorithm, we conducted a Mechanical Turk experiment that assessed the quality of suggested questions for news articles on celebrities. We compared our algorithm to a random baseline and to a partial version of our algorithm that includes a template relevance component but lacks filtering of candidate instantiations. The results show that the full algorithm provided 45% more correct instantiations, but surprisingly also 46% more relevant suggestions compared to the stronger baseline. These results point at the importance of both picking relevant templates and smart instantiation selection to the quality of generated questions. In addition, they indicate that user perception of relevance is affected by the correctness of the question.

## 2 Motivation and Algorithmic Overview

Before we detail our algorithm, we provide some motivations and insights to the design choices we took in our algorithm, which also indicate the difficulties inherent in the task.

### 2.1 Motivation

Given a news article, our algorithm generates a set of comparable questions for the article from question templates, e.g. “*who is faster #1 or #2?*”. Though the template words typically do not appear in the article, they need to be *relevant* to its content, that is they should correspond to one of the main themes in the article or to one of the pub-

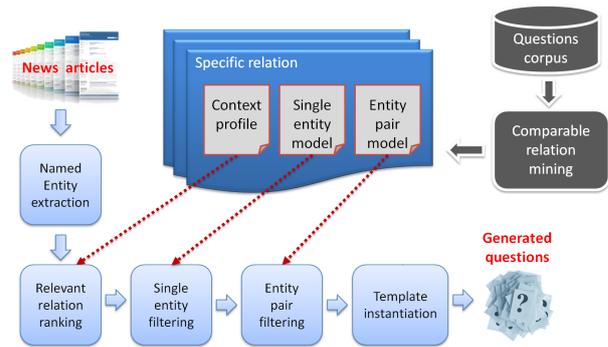


Figure 2: A high-level overview of the comparable question generation algorithm. The offline part is colored dark grey and the online part is colored light blue.

lic interests of the compared entities. For example, “*who is a better dad #1 or #2?*” is relevant to the article in Figure 1, while “*who is faster #1 or #2?*” is not relevant. Therefore, we need to model the typical contents to which each template is relevant.

Looking at the structure of comparable questions, we observed that a specific *comparable relation*, such as ‘*better dad*’ and ‘*faster*’, can usually be combined with named entities in several syntactic ways to construct a concrete question. We encode this information in *generic comparable templates*, e.g. “*who is a RE: #1 or #2?*” and “*is #1 a RE than #2?*”, where *RE* is a slot for a comparable relation and *#1* and *#2* are slots for entities. Using the above generic templates, ‘*Jet Li*’ and ‘*Jackie Chan*’ can be combined with the comparable relation ‘*better fighter*’ to generate “*who is a better fighter: Jackie Chan or Jet Li?*” and “*is Jackie Chan a better fighter than Jet Li?*” respectively. Following, our algorithm separately maintains comparable relations and generic templates.

In this paper we constrain ourselves to generate comparable questions between entities that appear in the article. Yet, not all entities can be compared to each other under a specific template, adding substantial complexity to the generation of questions. Looking at Figure 1, the generated question “*who is faster, Angelina Jolie or David Beckham?*” makes sense with respect to *David Beckham*, but not with respect to *Angelina Jolie*, since the typical reader is rarely interested in her running skills. Our algorithm thus needs to assess whether an instantiation is *correct*, that is whether the comparison between the two entities makes sense under the specific template.

Further delving into question correctness, the above example shows the need to assess each entity by itself. However, even if both entities are independently valid for the template, their comparison may not make sense. For example, “*who is better looking: Will Smith or Angelina Jolie?*” doesn’t feel right, even though each entity by itself fits the template. This is because when comparing looks, we expect a same sex comparison.

## 2.2 Algorithmic Overview

The above observations led us to the design of the automatic generation algorithm depicted in Figure 2. The algorithm’s offline part constructs, from a large collection of questions, a database of comparable relations, together with their typical contexts. It also extracts generic templates and the mapping to the relations that may instantiate them. From this database, we learn: (a) a *context profile* per template for relevance matching; (b) a *single entity model* per template slot that identify valid instantiations; and (c) an *entity pair model* that detects pairs of entities that can be compared together under the template. In the online part, these three models are applied to rank relevant templates for a given article and to generate only correct questions with respect to template instantiation.

The next two sections detail the template extraction component and the model training and application component in our algorithm.

## 3 Comparable Question Mining

To suggest comparable questions our algorithm needs a database of question templates. As discussed previously, a good source for mining such templates are CQA sites. Specifically, in this study we utilize all questions submitted to Yahoo! Answers in 2011 as our corpus. We next describe how comparable relations and generic comparable templates are extracted from this corpus.

### 3.1 Comparable Relation Extraction

An important observation for the task of comparable relation extraction is that many relations are complex multiword expressions, and thus their automatic detection is not trivial. Examples for such relations are marked in the questions “*Who is the best rapper alive, Eminem or Jay-z?*” and “*Who is the most beautiful woman in the world, Adriana Lima or Jessica Alba?*”. Therefore, we decided to employ a Conditional Random Fields (CRF) tag-

ger (Lafferty et al., 2001) to the task, since CRF was shown to be state-of-the-art for sequential relation extraction (Mooney and Bunescu, 2005; Culotta et al., 2006; Jindal and Liu, 2006).

As a pre-processing step for detecting comparable relations, our extraction algorithm identifies all the named entities of interest in our corpus, keeping only questions that contain at least two entities. In each of remaining questions, we then substitute the entity names with the variable slots #*i* in the order of their appearance. For example, “*Nnamdi Asomugha vs. Darrelle Revis? Who is the better cornerback?*” turned into “*#1 vs. #2? Who is the better cornerback?*”. This transformation helps us to design a simpler CRF than that of (Jindal and Liu, 2006), since our CRF utilizes the known positions of the target entities in the text.

To train the CRF model, the authors manually tagged all comparable relation words in approximately 300 transformed questions in the filtered corpus. The local and global features for the CRF, which we induce from each question word, are specified in Figures 3 and 4 respectively. Though there are many questions in Yahoo! Answers containing two named entities, e.g. “*Is #1 dating #2?*”, our CRF tagger is trained to detect only comparable relations like “*Who is prettier #1 or #2?*”. This is due to the labeled training set, which contains only this kind of relations, and to our features, which capture aspects of this specific linguistic structure.

The trained model was then applied to all other questions in the filtered corpus. This tagging process resulted in 60,000 identified question relation occurrences. From this output we constructed a database consisting of all occurring relations; each relation is accompanied by its *supporting questions*, those questions in which the relation occurrences were found. To achieve a highly accurate database, we filtered out relations with less than 50 supporting questions, ending with 295 relations in our database<sup>1</sup>. The authors conducted a manual evaluation of the CRF tagger performance, which showed 80% precision per occurrence. Yet, our filtering above of relations with low support left us with virtually 100% precision per relation and per occurrence.

<sup>1</sup>We intend to make this database publicly available under Yahoo! Webscope™ (<http://webscope.sandbox.yahoo.com>).

<sup>2</sup><http://nlp.stanford.edu/software/>

- (a) The word itself
- (b) Whether the word is capitalized
- (c) The word's suffixes of length 1,2, and 3, which helps in detecting comparative adjectives that ends 'est' or 'er'
- (d) The word's position in the sentence
- (e) The word's Part of speech (POS) tag, based on the Stanford POS tagger<sup>2</sup>
- (f) The words in a window of  $\pm 3$  around the current one
- (g) The adjective before the word, if exists, which helps detecting comparative noun phrases, e.g. 'better driver' and 'best singer'
- (h) The shortest word distance between the word and one of the  $\#i$  variables.
- (i) The shortest word distance of the word to one of the following connectives: 'between', 'out', ':', ',', '?'

Figure 3: CRF local features for each word

- (a) WH question type of the question, e.g. *what, which, who, where*
- (b) The average word distance between all  $\#i$  variables in the question
- (c) The conjunction tokens appearing between the  $\#i$  variables, such as *or, vs, and*

Figure 4: CRF global features for each word

### 3.2 Comparable Template Extraction

Our second mining task is to extract generic comparable templates that appear in our corpus, as well as identifying which comparable relation can instantiate which generic template.

To this end, we replace each recognized relation sequence with a variable  $RE$  in the support questions annotated with  $\#i$  variables. For example, "who is the best rapper alive,  $\#1$  or  $\#2$ ?" is transformed to "who is  $RE$ ,  $\#1$  or  $\#2$ ?". We next count the occurrences of each templated question. While some questions contain many details besides the comparable generic template, others are simpler and contain only the generic template. Through this counting, frequently occurring generic templates are revealed, such as "is  $\#1$  a  $RE$  than  $\#2$ ?". We retain only generic templates which appeared more than 50 times.

Finally, for each comparable relation we mark as applicable only generic templates that occur at least once in the supporting questions of this relation. For example, the template "who is  $RE$ :  $\#1$  or  $\#2$ ?" was found applicable for 'funnier', and thus could be used to generate the concrete question "who is funnier: Jennifer Aniston or Courteney Cox?". On average, each relation was associated with 3 generic templates.

---

### Algorithm 1 A high level overview of the online part of the question generation algorithm

---

**Input:** A news article

**Output:** A sorted list of comparable questions

- 1: Identify all target named entities (NEs) in the article
  - 2: Infer the distribution of LDA topics for the article
  - 3: For each comparable relation  $R$  in the database, compute its relevance score to be the similarity between the topic distributions of  $R$  and the article
  - 4: Rank all the relations according to their relevance score and pick the top  $M$  as relevant
  - 5: **for** each relevant relation  $R$  in the order of relevance ranking **do**
  - 6:     Filter out all the target NEs that do not pass the single entity classifier for  $R$
  - 7:     Generate all possible NE pairs from the those that passed the single classifier
  - 8:     Filter out all the generated NE pairs that do not pass the entity pair classifier for  $R$
  - 9:     Pick up the top  $N$  pairs with positive classification score to be qualified for generation
  - 10:    Instantiate  $R$  with each chosen NE pair via a randomly selected generic template
  - 11: **end for**
- 

## 4 Online Question Generation

The online part of our automatic generation algorithm takes as input a news article and generates concrete comparable questions for it. Its high level description is presented in Algorithm 1. The algorithm starts with identifying the comparable relations in our database that are relevant to the article. For each relevant relation, we then generate concrete questions by picking generic templates that are applicable for this relation and instantiating them with pairs of named entities appearing in the article. Yet, as discussed before, only for some entity pairs the comparison under the specific relation makes sense, a quality which we refer to as instantiation correctness (see Section 2). To this end, we utilize two supervised models to filter incorrect instantiations. We next detail the two aspects of the online part: ranking relevant relations and correctly instantiating relations.

### 4.1 Ranking relevant relations

To assess how relevant a given comparable relation is to an article, we model the relation's typical context as a distribution over latent semantic topics. Specifically, we utilize Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to infer latent topics in texts.

To train an LDA model, we constructed for each comparable relation a pseudo-document consisting of all questions that contain this relation in our corpus (the supporting questions). We then

trained a model of 200 topics over these pseudo-documents, resulting in a model over a lexicon of 107,835 words. An additional product of the LDA training process is a topic distribution for each relation’s pseudo-document, which we consider as the relation’s *context profile*. We note that, unless otherwise specified, different model parameters were chosen based on a small held out collection of articles and questions, manually annotated by the authors. This collection was used to validate that the chosen parameter values indeed “make sense” for the task.

Given a news article, a distribution over LDA topics is inferred from the article’s text using the trained model. Then, a cosine similarity between this distribution and the context profile of each comparable relation in our database is computed and taken as the relevance score for this relation. Finally, we rank all relations according to their relevance score and pick the top  $M$  as candidates for instantiation ( $M=3$  in our experiment).

## 4.2 Correctly instantiating relations

To generate useful questions from relevant comparable relations, we need to retain only correct instantiations of these relations. To this end, we utilize two complementing types of filters, one for each entity by itself, and one for pairs, since each filter considers different attributes of the entities at hand. For example, for the relation ‘*is faster*’, the single entity filter looks for athletes of all kinds, for whom this comparison is of interest to the reader. The pair filter, on the other hand, attempts to pass only same sex and same profession comparisons, *e.g.* male football players or female baseball players for this relation.

We next describe the various features we extract for every entity and the supervised models that given this feature vector representation assess the correctness of an instantiation.

### 4.2.1 Entity Features

We want to represent each entity as a vector of features that capture different aspects of entity characterization. To this end, we utilize two different broad-scale sources of information about named entities. The first is DBPedia<sup>3</sup>, which contains structured information on entries in Wikipedia, many of them are named entities that appear in news articles. The second source is the corpus of

<sup>3</sup><http://wiki.dbpedia.org/About>

CQA questions, which in our study was harvested from Yahoo! Answers (see Section 3).

For named entities with a DBPedia entry, we extract all the DBPedia properties of classes *subject* and *type* as indicator features. Some example features for *Brad Pitt* include *Actors\_from\_Oklahoma*, *AmericanAtheists*, *Artist* and *American\_film\_producers*.

One property that is currently missing from DBPedia is *gender*, a feature that was found to be very useful in our experiments. We automatically induce this feature from the Wikipedia abstract in each DBPedia entry. Specifically, we construct a histogram of male and female pronouns: *he* and *his* vs. *she* and *her*. The majority pronoun sex is then chosen to be the gender of the named entity, or none if the histogram is empty.

One way to utilize the CQA question corpus could be to extract co-occurring words with each target entity as relevant contexts. Yet, since our questions come from Yahoo! Answers, we decided to use another attribute of the questions, the category to which the question is assigned, within a hierarchy of 1,669 categories (*e.g.* ‘*Sports>Baseball*’ and ‘*Pets>Dogs*’). For each named entity, we construct a histogram of the number of questions containing it that are assigned to each category. This histogram is normalized into a probability distribution with Laplace smoothing of 0.03, to incorporate the uncertainty that lies in named entities that appear only very few times. The categories and their probabilities are added as features, providing a high level representation of relevant contexts for the entity.

### 4.2.2 Single entity filtering

We view the task of single entity filtering as a classification task. To this end, we trained a classifier per relation, constructing a different labeled training set for each relation. Positive examples are the entities that instantiate this relation in our CQA corpus. As negative examples, we take named entities that were never seen instantiating the relation in the corpus, but still occurred in some questions. We note that our named entity tagger could recognize more than 200,000 named entities, and most of them are negative for a given relation.

For each relation we select negative examples by sampling uniformly from its negative entity list, assuming that the probability of hitting false negatives is low for such a long list. It is known that better classification performance is typically

achieved for a balanced training set (Provost, 2000). In our case, we over sample to help the classifier explore the large space of negative examples. Specifically, we sample 2,000 negative examples and duplicate the positive set to reach a similar number.

We utilize the Support Vector Machines (SVM) implementation of LIBSVM (Chang and Lin, 2011) with a linear kernel as our classifier. The feature vector of each named entity was induced as described in Section 4.2.1. We split the labeled dataset into 70% training set and 30% validation set. Feature selection using information gain was performed on the training set to throw out non-significant features (Mitchell, 1997). The average accuracy of the single classifiers, measured over the validation sets, was 91%.

### 4.2.3 Entity pair filtering

Similar to single entity filtering, we view the task of filtering entity pairs as a classification task, training a separate classifier for each relation. Entity pairs that instantiate the given relation in the question corpus are considered positive examples. Yet, the space of all the pairs that never instantiated the relation is huge, and the set of positive examples is relatively much smaller compared to the situation in the single entity classifier. In our study, uniform negative example sampling turned the training into a trivial task, preventing from the classifier to choose an useful discriminative boundary. Therefore, we generate negative examples by sampling only from pairs of named entities that both pass the single entity filter for this relation. The risk here is that we may sample false negative examples. Still, this sampling scheme enabled the classifier to identify better discriminative features.

To generate features for a candidate pair, we take the two feature vectors of the two entities and induce families of pair features by comparing between the two vectors. Figure 5 describes the various features we generate. We utilize LIBSVM with an RBF kernel for this task, splitting the examples into 70% training set and 30% validation set. We over sampled the positive examples to reach up to 100 examples.

The average accuracy of the pair classifiers on the validation set was 83%. For example, named entities that pass the single entity filtering for “*be funny*”, include Jay Leno, David Letterman (American TV hosts), Jim Carrey, and Steve Mar-

- (a) All shared DBPedia indicator features in the two vectors:  $f_a^{DBPedia} \cap f_b^{DBPedia}$ , indicating them as shared, e.g. ‘*FilmMaker\_s*’
  - (b) All DBPedia features that appear only in one of the vectors, termed *one-side features*:  $f_a^{DBPedia} \setminus f_b^{DBPedia}$  and  $f_b^{DBPedia} \setminus f_a^{DBPedia}$ , indicating them as such, e.g. ‘*FilmMaker\_o*’
  - (c) Wikipedia categories that are ancestors of at least two one-side features that appear in the training set. For example, a common ancestor of ‘*Spanish\_actors*’ and ‘*Russian\_actors*’ is ‘*European\_actors*’. These features provide a high level perspective on one-side features
  - (d) The Yahoo! Answers categories in which both named entities appear
  - (e) Hellinger distance (Pollard, 2001) between the probability distributions over categories of the two entities
  - (f) Three indicator gender features: whether both named entities are males, both are females or are different

Figure 5: The entity pair features generated from two single entity feature vectors  $f_a$  and  $f_b$

tin (actors). The pair classifier assigned positive scores only to {*Jay Leno, David Letterman*} (TV hosts) and {*Jim Carrey, Steve Martin*} (actors) but not to other pairings of these entities.

## 5 Evaluation

### 5.1 Experimental Settings

To evaluate our algorithm’s performance, we designed a Mechanical Turk (MTurk) experiment in which human annotators assess the quality of the questions that our algorithm generates for a sample of news articles. As the source of test articles, we chose the OMG! website<sup>4</sup>, which contains news articles on celebrities.

Test articles were selected by first randomly sampling 5,000 news article from those that were posted on OMG! in 2011. We then filtered out articles that are longer than 4,000 characters, which were found to be tiresome for annotators to read, and those that are shorter than 300 characters, which consist mainly of video and photos. We were left with a pool of 1,016 articles from which we randomly sampled 100 as the test set.

For each test article our algorithm obtained the top three relevant comparable relations, and for each relation selected the best instantiation (if exists). We used two baselines for performance comparison. The first *random baseline* chooses a relation randomly out of all possible relations in the database and then instantiates it with a random pair of entities that appear in the article. The second *relevance baseline* chooses the most relevant

<sup>4</sup><http://www.omg.com/>

	Relevance	Correctness
Random baseline	29%	43%
Relevance baseline	37%	53%
Full algorithm	54%	77%

Table 1: Relevance and correctness percentage by tested algorithm

relation to the article based on our algorithm, but still instantiates it with a random pair. For each test article, we presented to the evaluators the questions generated by the three tested algorithms in a random order to avoid any bias. We note that our second baseline enabled us to measure the stand-alone contribution of the LDA-based relevance model. In addition, it enabled us to measure the relative contribution of the instantiation models on top of relevance model.

Each article was evaluated by 10 MTurk workers, which were asked to mark for each displayed question whether it is relevant and whether it is correct (see Section 2 for relevance and correctness definitions). The workers were given precise instructions along with examples before they started the test. A control story was used to filter out dishonest or incapable workers<sup>5</sup>.

## 5.2 Results

For each tested algorithm, we separately counted the percentage of annotations that marked each question as relevant and the percentage of annotations that marked each question as instantiated correctly, denoted *relevance score* and *correctness score*. We then averaged these scores over all questions that were displayed for the test articles. The results are presented in Table 1. The differences between the full algorithm and the baselines are statistically significant at  $p < 0.01$  and between baselines the differences are statistically significant at  $p < 0.05$  using the Wilcoxon double-sided signed-ranks test (Wilcoxon, 1945).

Our main result is that our full algorithm substantially outperforms the stronger relevance baseline. It improves the correctness score by 45%, which points at the effectiveness of our two step filtering of incorrect instantiations. It’s performance is just under 80%, showing high quality entity pair selection for relations. Yet, we did not expect to see an increase of 46% in the relevance

<sup>5</sup>We intend to make the tested articles, the instructions to annotators and their annotations publicly available under Yahoo! Webscope™ (<http://webscope.sandbox.yahoo.com>).

metric, since both the full algorithm and the relevance baseline use the same relevance component to rank relations by. One explanation for this is that sometimes the instantiation filter eliminates all possible entity pairs for some relation that is incorrectly considered relevant by the algorithm. Thus, the filtering of entities provides also an additional filtering perspective on relevance. In addition, it may be that humans tend to be more permissive when assessing the relevance of a correctly instantiated question.

To illustrate the differences between baselines and the full algorithm, Table 2 presents an example article together with the suggested generated questions by each algorithm. The random baseline picked an irrelevant relation, and while the relevance baseline selected a relevant relation, “*a better president*”, it was instantiated incorrectly. The full algorithm, on the other hand, both chose relevant relations for all three questions and instantiated them correctly. Especially, the incorrectly instantiated relation in the relevance baseline is now correctly instantiated with plausible presidential candidates.

Comparing between baselines, the relevance baseline beats the random baseline by 28% in terms of relevance. This is not surprising, since this was the focus of this baseline. Yet, it also improved correctness by 23% over the random baseline. This is an unexpected result that indicates that when users view relevant relations, they may be more forgiving in their perception of unreasonable instantiations.

For each article, our full algorithm attempts to generate three questions, one for each of the top three relevant questions. It is possible that for some articles not all three questions will be generated, due to instantiation filtering. We found that for 85% of the articles all three questions were generated. For the remaining 15% at least one question was always generated, and for  $\frac{1}{3}$  of them two questions were composed. Furthermore, we found that the relevance and correctness scores were not affected by the position of the question. In the case of instantiation correctness, since the best pair was picked for each relation and this component is quite accurate, this is somewhat expected. In the case of relevance, this indicates that there are usually several relations in our database that are relevant to the article.

Ron Livingston is teaming up with Tom Hanks and HBO again after their successful 2001 collaboration on Band of Brothers. The actor has been cast in HBO's upcoming film Game Change that centers on the 2008 presidential campaign, Deadline reports. He joins Ed Harris, Julianne Moore and Woody Harrelson. The Jay Roach-directed movie follows John McCain (Harris) as he selects Alaska Gov. Sarah Palin (Moore) as his running mate, throughout the campaign and to their ultimate defeat to Barack Obama. Livingston will play Mark Wallace, one of the campaign's senior advisors and the man who prepped Palin for her debate. Harrelson will play campaign strategist Steve Schmidt. . .

Algorithm	Question
Random baseline	Who is a better singer, Sarah Palin or Barack Obama ?
Relevance baseline	Would Ron Livingston be a better president than Julianne Moore ?
Full algorithm	Who has the best movies Tom Hanks or Julianne Moore ?
Full algorithm	Is John Mccain a better leader than Barack Obama ?
Full algorithm	Would Sarah Palin be a better president than John Mccain ?

Table 2: Automatically generated questions by the baselines and the full algorithm to an example article

### 5.3 Error Analysis

To better understand the performance of our algorithm, we looked at some low quality questions that were generated, either due to incorrect instantiation or due to irrelevance to the article.

Starting with relevance, one of the repeating mistakes was promoting relations that are related to a list of named entities in the article, but not to its main theme. For example, the relation ‘*who is a better actor*’ was incorrectly ranked high for an article about Ricky Gervais claiming that he has been asked to host Globes again after he offended Angelina Jolie, Johnny Depp, Robert Downey Jr. and Charlie Sheen, among others during last Globes ceremony. The reason for this mistake is that many named entities appear as frequent terms in LDA topics, and thus mentioning many names that belong to a single topic drives LDA to assign this topic a high probability. Yet, unlike other cases, here entity filtering does not help ignoring such errors, since the same entities that triggered the ranking of the relation are also valid instantiations for it.

Analyzing incorrect instantiations, many mistakes are due to mismatches between the two compared entities that were too fine grained for our algorithm to catch. For example, “*who’s the better guitarist: Paul McCartney or Ringo Starr?*” was generated since our algorithm failed to identify that *Ringo Starr* is a drummer rather than a guitarist, though both participants in the relation are musicians. In other cases, strong co-occurrence of the two celebs in our question corpus convinced the classifiers that they can be matched. For example, “*who is a better dancer Michael Jackson or Debbie Rowe?*” was incorrectly generated, since Debbie Rowe is not a dancer. Yet, she was Michael Jackson’s wife and they appear together

in a lot of questions in our corpus.

## 6 Related Work

Traditionally, question generation focuses on converting assertions in a text into question forms (Brown et al., 2005; Mitkov et al., 2006; Myller, 2007; Heilman and Smith, 2010; Rus et al., 2010; Agarwal et al., 2011; Olney et al., 2012). To the best of our knowledge, there is no prior work on our task, which is to generate relevant synthetic questions whose content, except for the arguments, might not appear in the text.

Our extraction of comparable relations falls within the field of Relation Extraction, in which CRF is a state-of-the-art method (Mooney and Bunescu, 2005; Culotta et al., 2006). We note that in the works of Jindal and Liu (2006) and Li et. al. (2010) comparative questions are identified as an intermediate step for the task of extracting compared entities, which are unknown in their setting. We, on the other hand, detect the compared entities in a pre-processing step, and our target is the extraction of the comparable relations given known candidate entities.

Our algorithm ranks relevant templates based on the similarity between an article’s content and the typical context of each relation. Prior work rank relevant concrete questions to a given input question, focusing on strong lexical similarities (Jeon et al., 2005; Cai et al., 2011; Hao and Agichtein, 2012). We, however, do not expect to find direct lexical similarities between candidate relations and the article. Instead, we are interested in a higher level topical similarity to the input article, for which LDA topics were shown to help (Celikyilmaz et al., 2010).

Finally, several works present unsupervised methods for ranking proper template instantia-

tions, mainly as selectional preferences (Light and Greiff, 2002; Erk, 2007; Ritter et al., 2010). However, we eventually choose instantiation candidates, and thus preferred supervised methods that enable filtering and not just ranking. Furthermore, we target a more subtle discrimination between entities than prior work, *e.g.* between quarterbacks, singers and actors. Machine learning naturally incorporates the many features that capture different aspects of entity characterization.

## 7 Conclusions

We introduced the novel task of automatically generating synthetic comparable questions that are relevant to a given news article but do not necessarily appear in it. To this end, we proposed an algorithm that consists of two parts. The offline part identifies comparable relations in a large collection of questions. Its output is a database of comparable relations together with a context profile for each relation and models that detect correct instantiations of this relation, all learned from the question corpus. In the online part, given a news article, the algorithm identifies relevant comparable relations based on the similarity between the article content and each relation’s context profile. Then, relevant relations are instantiated only with pairs of named entities from the article whose comparison makes sense by applying the instantiation correctness models to candidate pairs.

We assessed the performance of our algorithm via a Mechanical Turk experiment. A partial version of our algorithm, without instantiation filtering, was our strongest baseline. The full algorithm outperformed this baseline by 45% on question correctness, but surprisingly also by 46% on question relevance. These results show that our supervised filtering methods are successful in keeping only correct pairs, but they also serve as an additional filtering for relevant relations, on top of context matching.

In future work, we want to generate more diverse and intriguing questions by selecting relevant named entities for template instantiation that do not appear in the article. Another direction would be take a supervised approach, training classifiers over a labeled dataset for filtering irrelevant templates and incorrect instantiations. Finally, it would be interesting to see how our algorithm performs on other news domains.

## References

- Manish Agarwal, Rakshit Shah, and Prashanth Manem. 2011. Automatic question generation using discourse cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications, IUNLPBEA ’11*, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.
- Jonathan C. Brown, Gwen A. Frishkoff, and Maxine Eskenazi. 2005. Automatic question generation for vocabulary assessment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT ’05*, pages 819–826, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. 2011. Learning the latent topics for question retrieval in community qa. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 273–281, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Asli Celikyilmaz, Dilek Hakkani-Tur, and Gokhan Tur. 2010. Lda based similarity modeling for question answering. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search, SS ’10*, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chih-Chung Chang and Chih-Jen Lin. 2011. Libsvm: A library for support vector machines. *ACM TIST*, 2(3):27.
- Aron Culotta, Andrew McCallum, and Jonathan Betz. 2006. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, HLT-NAACL ’06*, pages 296–303, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 216–223, Prague, Czech Republic, June. Association for Computational Linguistics.
- Tianyong Hao and Eugene Agichtein. 2012. Finding similar questions in collaborative question answering archives: toward bootstrapping-based equivalent pattern learning. *Inf. Retr.*, 15(3-4):332–353, June.
- Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation.

- In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 609–617, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 84–90, New York, NY, USA. ACM.
- Nitin Jindal and Bing Liu. 2006. Mining comparative sentences and relations. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1331–1336. AAAI Press.
- John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*.
- Shasha Li, Chin-Yew Lin, Young-In Song, and Zhoujun Li. 2010. Comparable entity mining from comparative questions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 650–658. Association for Computational Linguistics.
- Marc Light and Warren R. Greiff. 2002. Statistical models for the induction and use of selectional preferences. *Cognitive Science*, 26(3):269–281.
- Tom M. Mitchell. 1997. *Machine learning*. McGraw Hill series in computer science. McGraw-Hill.
- Ruslan Mitkov, Le An Ha, and Nikiforos Karamanis. 2006. A computer-aided environment for generating multiple-choice test items. *Nat. Lang. Eng.*, 12(2):177–194, June.
- Raymond J. Mooney and Razvan Bunescu. 2005. Mining knowledge from text using information extraction. *SIGKDD Explor. Newsl.*, 7(1):3–10, June.
- Niko Myller. 2007. Automatic generation of prediction questions during program visualization. *Electron. Notes Theor. Comput. Sci.*, 178:43–49, July.
- A.M. Olney, A.C. Graesser, and N.K. Person. 2012. Question generation from concept maps. *Dialogue & Discourse*, 3(2):75–99.
- D. Pollard. 2001. *A User's Guide to Measure Theoretic Probability*. Cambridge University Press.
- F. Provost. 2000. Machine learning from imbalanced data sets 101. *Proceedings of the AAAI-2000 Workshop on Imbalanced Data Sets*.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 424–434, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Vasile Rus, Brendan Wyse, Paul Piwek, Mihai C. Lindean, Svetlana Stoyanchev, and Cristian Moldovan. 2010. The first question generation shared task evaluation challenge. In John D. Kelleher, Brian Mac Namee, Ielka van der Sluis, Anja Belz, Albert Gatt, and Alexander Koller, editors, *INLG 2010 - Proceedings of the Sixth International Natural Language Generation Conference, July 7-9, 2010, Trim, Co. Meath, Ireland*. The Association for Computer Linguistics.
- Anne Schuth, Maarten Marx, and Maarten de Rijke. 2007. Extracting the discussion structure in comments on news-articles. In *Proceedings of the 9th annual ACM international workshop on Web information and data management*, WIDM '07, pages 97–104, New York, NY, USA. ACM.
- Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1:80–83.

# Punctuation Prediction with Transition-based Parsing

Dongdong Zhang<sup>1</sup>, Shuangzhi Wu<sup>2</sup>, Nan Yang<sup>3</sup>, Mu Li<sup>1</sup>

<sup>1</sup>Microsoft Research Asia, Beijing, China

<sup>2</sup>Harbin Institute of Technology, Harbin, China

<sup>3</sup>University of Science and Technology of China, Hefei, China

{dozhang, v-shuawu, v-nayang, muli}@microsoft.com

## Abstract

Punctuations are not available in automatic speech recognition outputs, which could create barriers to many subsequent text processing tasks. This paper proposes a novel method to predict punctuation symbols for the stream of words in transcribed speech texts. Our method jointly performs parsing and punctuation prediction by integrating a rich set of syntactic features when processing words from left to right. It can exploit a global view to capture long-range dependencies for punctuation prediction with linear complexity. The experimental results on the test data sets of IWSLT and TDT4 show that our method can achieve high-level performance in punctuation prediction over the stream of words in transcribed speech text.

## 1 Introduction

Standard automatic speech recognizers output unstructured streams of words. They neither perform a proper segmentation of the output into sentences, nor predict punctuation symbols. The unavailable punctuations and sentence boundaries in transcribed speech texts create barriers to many subsequent processing tasks, such as summarization, information extraction, question answering and machine translation. Thus, the segmentation of long texts is necessary in many real applications. For example, in speech-to-speech translation, continuously transcribed speech texts need to be segmented before being fed into subsequent machine translation systems (Takezawa et al., 1998; Nakamura, 2009). This is because current machine translation (MT) systems perform the translation at the sentence level, where various models used in MT are trained over segmented sentences and many algorithms inside MT have an exponential complexity with regard to the length of inputs.

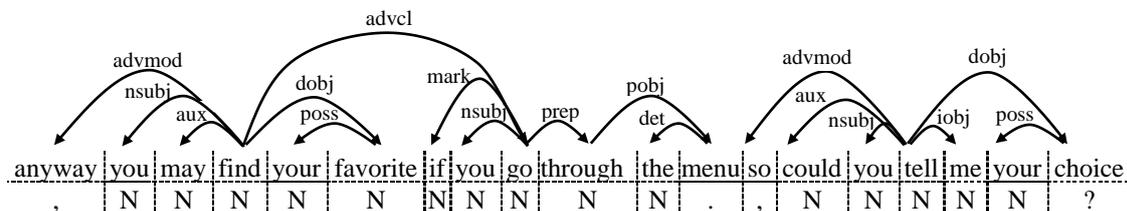
The punctuation prediction problem has attracted research interest in both the speech pro-

cessing community and the natural language processing community. Most previous work primarily exploits local features in their statistical models such as lexicons, prosodic cues and hidden event language model (HELM) (Liu et al., 2005; Matusov et al., 2006; Huang and Zweig, 2002; Stolcke and Shriberg, 1996). The word-level models integrating local features have narrow views about the input and could not achieve satisfied performance due to the limited context information access (Favre et al., 2008). Naturally, global contexts are required to model the punctuation prediction, especially for long-range dependencies. For instance, in English question sentences, the ending question mark is long-range dependent on the initial phrases (Lu and Ng, 2010), such as “*could you*” in Figure 1. There has been some work trying to incorporate syntactic features to broaden the view of hypotheses in the punctuation prediction models (Roark et al., 2006; Favre et al., 2008). In their methods, the punctuation prediction is treated as a separated post-procedure of parsing, which may suffer from the problem of error propagation. In addition, these approaches are not able to incrementally process inputs and are not efficient for very long inputs, especially in the cases of long transcribed speech texts from presentations where the number of streaming words could be larger than hundreds or thousands.

In this paper, we propose jointly performing punctuation prediction and transition-based dependency parsing over transcribed speech text. When the transition-based parsing consumes the stream of words left to right with the shift-reduce decoding algorithm, punctuation symbols are predicted for each word based on the contexts of the parsing tree. Two models are proposed to cause the punctuation prediction to interact with the transition actions in parsing. One is to conduct transition actions of parsing followed by punctuation predictions in a cascaded way. The other is to associate the conventional transition actions of parsing with punctuation predictions, so that predicted punctuations are directly inferred from the

anyway you may find your favorite if you go through the menu so could you tell me your choice

(a). The transcribed speech text without punctuations



(b). Transition-based parsing trees and predicted punctuations over transcribed text

anyway, you may find your favorite if you go through the menu. so, could you tell me your choice?

(c). Two segmentations are formed when inserting the predicted punctuation symbols into the transcribed text

Figure 1. An example of punctuation prediction.

parsing tree. Our models have linear complexity and are capable of handling streams of words with any length. In addition, the computation of models use a rich set of syntactic features, which can improve the complicated punctuation predictions from a global view, especially for the long range dependencies.

Figure 1 shows an example of how parsing helps punctuation prediction over the transcribed speech text. As illustrated in Figure 1(b), two commas are predicted when their preceding words act as the adverbial modifiers (*advmod*) during parsing. The period after the word “*menu*” is predicted when the parsing of an adverbial clause modifier (*advcl*) is completed. The question mark at the end of the input is determined when a direct object modifier (*dobj*) is identified, together with the long range clue that the auxiliary word occurs before the nominal subject (*nsubj*). Eventually, two segmentations are formed according to the punctuation prediction results, shown in Figure 1(c).

The training data used for our models is adapted from Treebank data by excluding all punctuations but keeping the punctuation contexts, so that it can simulate the unavailable annotated transcribed speech texts. In decoding, beam search is used to get optimal punctuation prediction results. We conduct experiments on both IWSLT data and TDT4 test data sets. The experimental results show that our method can achieve higher performance than the CRF-based baseline method.

The paper is structured as follows: Section 2 conducts a survey of related work. The transition-based dependency parsing is introduced in Section

3. We explain our approach to predicting punctuations for transcribed speech texts in Section 4. Section 5 gives the results of our experiment. The conclusion and future work are given in Section 6.

## 2 Related Work

Sentence boundary detection and punctuation prediction have been extensively studied in the speech processing field and have attracted research interest in the natural language processing field as well. Most previous work exploits local features for the task. Kim and Woodland (2001), Huang and Zweig (2002), Christensen et al. (2001), and Liu et al. (2005) integrate both prosodic features (pitch, pause duration, etc.) and lexical features (words, n-grams, etc.) to predict punctuation symbols during speech recognition, where Huang and Zweig (2002) uses a maximum entropy model, Christensen et al. (2001) focus on finite state and multi-layer perceptron methods, and Liu et al. (2005) uses conditional random fields. However, in some scenarios the prosodic cues are not available due to inaccessible original raw speech waveforms. Matusov et al. (2006) integrate segmentation features into the log-linear model in the statistical machine translation (SMT) framework to improve the translation performance when translating transcribed speech texts. Lu and Ng (2010) uses dynamic conditional random fields to perform both sentence boundary and sentence type prediction. They achieved promising results on both English and Chinese transcribed speech texts. The above work only ex-

exploits local features, so they were limited to capturing long range dependencies for punctuation prediction.

It is natural to incorporate global knowledge, such as syntactic information, to improve punctuation prediction performance. Roark et al. (2006) use a rich set of non-local features including parser scores to re-rank full segmentations. Favre et al. (2008) integrate syntactic information from a PCFG parser into a log-linear and combine it with local features for sentence segmentation. The punctuation prediction in these works is performed as a post-procedure step of parsing, where a parse tree needs to be built in advance. As their parsing over the stream of words in transcribed speech text is exponentially complex, their approaches are only feasible for short input processing. Unlike these works, we incorporate punctuation prediction into the parsing which process left to right input without length limitations.

Numerous dependency parsing algorithms have been proposed in the natural language processing community, including transition-based and graph-based dependency parsing. Compared to graph-based parsing, transition-based parsing can offer linear time complexity and easily leverage non-local features in the models (Yamada and Matsumoto, 2003; Nivre et al., 2006b; Zhang and Clark, 2008; Huang and Sagae, 2010). Starting with the work from (Zhang and Nivre, 2011), in this paper we extend transition-based dependency parsing from the sentence-level to the stream of words and integrate the parsing with punctuation prediction.

Joint POS tagging and transition-based dependency parsing are studied in (Hatori et al., 2011; Bohnet and Nivre, 2012). The improvements are reported with the joint model compared to the pipeline model for Chinese and other richly inflected languages, which shows that it also makes sense to jointly perform punctuation prediction and parsing, although these two tasks of POS tagging and punctuation prediction are different in two ways: 1). The former usually works on a well-formed single sentence while the latter needs to process multiple sentences that are very lengthy. 2). POS tags are must-have features to parsing while punctuations are not. The parsing quality in the former is more sensitive to the performance of the entire task than in the latter.

### 3 Transition-based dependency parsing

In a typical transition-based dependency parsing process, the shift-reduce decoding algorithm is

applied and a queue and stack are maintained (Zhang and Nivre, 2011). The queue stores the stream of transcribed speech words, the front of which is indexed as the current word. The stack stores the unfinished words which may be linked with the current word or a future word in the queue. When words in the queue are consumed from left to right, a set of transition actions is applied to build a parse tree. There are four kinds of transition actions conducted in the parsing process (Zhang and Nivre, 2011), as described in Table 1.

Action	Description
<i>Shift</i>	Fetches the current word from the queue and pushes it to the stack
<i>Reduce</i>	Pops the stack
<i>LeftArc</i>	Adds a dependency link from the current word to the stack top, and pops the stack
<i>RightArc</i>	Adds a dependency link from the stack top to the current word, takes away the current word from the queue and pushes it to the stack

Table 1. Action types in transition-based parsing

The choice of each transition action during the parsing is scored by a linear model that can be trained over a rich set of non-local features extracted from the contexts of the stack, the queue and the set of dependency labels. As described in (Zhang and Nivre, 2011), the feature templates could be defined over the lexicons, POS-tags and the combinations with syntactic information.

In parsing, beam search is performed to search the optimal sequence of transition actions, from which a parse tree is formed (Zhang and Clark, 2008). As each word must be pushed to the stack once and popped off once, the number of actions needed to parse a sentence is always  $2n$ , where  $n$  is the length of the sentence. Thus, transition-based parsing has a linear complexity with the length of input and naturally it can be extended to process the stream of words.

## 4 Our method

### 4.1 Model

In the task of punctuation prediction, we are given a stream of words from an automatic transcription of speech text, denoted by  $w_1^n := w_1, w_2, \dots, w_n$ . We are asked to output a sequence of punctuation symbols  $S_1^n := s_1, s_2, \dots, s_n$  where  $s_i$  is attached to  $w_i$  to form a sentence like Figure 1(c). If there are no ambiguities,  $S_1^n$  is also abbreviated as  $S$ ,

similarly for  $w_1^n$  as  $w$ . We model the search of the best sequence of predicted punctuation symbols  $S^*$  as:

$$S^* = \operatorname{argmax}_S P(S_1^n | w_1^n) \quad (1)$$

We introduce the transition-based parsing tree  $T$  to guide the punctuation prediction in Model (2), where parsing trees are constructed over the transcribed text while containing no punctuations.

$$S^* = \operatorname{argmax}_S \sum_T P(T | w_1^n) \times P(S_1^n | T, w_1^n) \quad (2)$$

Rather than enumerate all possible parsing trees, we jointly optimize the punctuation prediction model and the transition-based parsing model with the form:

$$(S^*, T^*) = \operatorname{argmax}_{(S,T)} P(T | w_1^n) \times P(S_1^n | T, w_1^n) \quad (3)$$

Let  $T_1^i$  be the constructed partial tree when  $w_1^i$  is consumed from the queue. We decompose the Model (3) into:

$$(S^*, T^*) = \operatorname{argmax}_{(S,T)} \prod_{i=1}^n P(T_1^i | T_1^{i-1}, w_1^i) \times P(s_i | T_1^i, w_1^i) \quad (4)$$

It is noted that a partial parsing tree uniquely corresponds to a sequence of transition actions, and vice versa. Suppose  $T_1^i$  corresponds to the action sequence  $A_1^i$  and let  $a_i$  denote the last action in  $A_1^i$ . As the current word  $w_i$  can only be consumed from the queue by either *Shift* or *RightArc* according to Table 1, we have  $a_i \in \{\textit{Shift}, \textit{RightArc}\}$ . Thus, we synchronize the punctuation prediction with the application of *Shift* and *RightArc* during the parsing, which is explained by Model (5).

$$(S^*, T^*) = \operatorname{argmax}_{(S,T)} \prod_{i=1}^n P(T_1^i, A_1^i | T_1^{i-1}, w_1^i) \times P(s_i | a_i, T_1^i, w_1^i) \quad (5)$$

The model is further refined by reducing the computation scope. When a full-stop punctuation is determined (i.e., a segmentation is formed), we discard the previous contexts and restart a new

procedure for both parsing and punctuation prediction over the rest of words in the stream. In this way we are theoretically able to handle the unlimited stream of words without needing to always keep the entire context history of streaming words. Let  $b_i$  be the position index of last full-stop punctuation<sup>1</sup> before  $i$ ,  $T_{b_i}^i$  and  $A_{b_i}^i$  the partial tree and corresponding action sequence over the words  $w_{b_i}^i$ , Model (5) can be rewritten by:

$$(S^*, T^*) = \operatorname{argmax}_{(S,T)} \prod_{i=1}^n P(T_{b_i}^i, A_{b_i}^i | T_{b_i}^{i-1}, w_{b_i}^i) \times P(s_i | a_i, T_{b_i}^i, w_{b_i}^i) \quad (6)$$

With different computation of Model (6), we induce two joint models for punctuation prediction: the cascaded punctuation prediction model and the unified punctuation prediction model.

## 4.2 Cascaded punctuation prediction model (CPP)

In Model (6), the computation of two sub-models is independent. The first sub-model is computed based on the context of words and partial trees without any punctuation knowledge, while the computation of the second sub-model is conditional on the context from the partially built parsing tree  $T_{b_i}^i$  and the transition action. As the words in the stream are consumed, each computation of transition actions is followed by a computation of punctuation prediction. Thus, the two sub-models are computed in a cascaded way, until the optimal parsing tree and optimal punctuation symbols are generated. We call this model the *cascaded punctuation prediction model (CPP)*.

## 4.3 Unified punctuation prediction model (UPP)

In Model (6), if the punctuation symbols can be deterministically inferred from the partial tree,  $P(s_i | a_i, T_{b_i}^i, w_{b_i}^i)$  can be omitted because it is always 1. Similar to the idea of joint POS tagging and parsing (Hatori et al., 2011; Bohnet and Nivre, 2012), we propose attaching the punctuation prediction onto the parsing tree by embedding  $s_i$  into  $a_i$ . Thus, we extend the conventional transition actions illustrated in Table 1 to a new set of transition actions for the parsing, denoted by  $\hat{A}$ :

<sup>1</sup> Specially,  $b_i$  is equal to 1 if there are no previous full-stop punctuations.

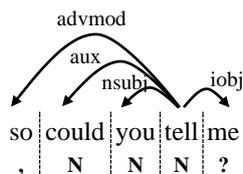
$$\hat{A} = \{LeftArc, Reduce\} \cup \{Shift(s) | s \in Q\} \\ \cup \{RightArc(s) | s \in Q\}$$

where  $Q$  is the set of punctuation symbols to be predicted,  $s$  is a punctuation symbol belonging to  $Q$ ,  $Shift(s)$  is an action that attaches  $s$  to the current word on the basis of original  $Shift$  action in parsing,  $RightArc(s)$  attaches  $s$  to the current word on the basis of original  $RightArc$  action.

With the redefined transition action set  $\hat{A}$ , the computation of Model (6) is reformulated as:

$$(S^*, T^*) = \operatorname{argmax}_{(S, T)} \prod_{i=1}^n P \left( T_{b_i}^i, \hat{A}_{b_i}^i \mid T_{b_i}^{i-1}, \hat{A}_{b_i}^{i-1}, w_{b_i}^i \right) \quad (7)$$

Here, the computation of parsing tree and punctuation prediction is unified into one model where the sequence of transition action outputs uniquely determines the punctuations attached to the words. We refer to it as the **unified punctuation prediction model (UPP)**.



(a). Parsing tree and attached punctuation symbols

$Shift(,)$ ,  $Shift(N)$ ,  $Shift(N)$ ,  $LeftArc$ ,  $LeftArc$ ,  $LeftArc$ ,  $Shift(N)$ ,  $RightArc(?)$ ,  $Reduce$ ,  $Reduce$

(b). The corresponding sequence of transition actions

Figure 2. An example of punctuation prediction using the UPP model, where N is a null type punctuation symbol denoting no need to attach any punctuation to the word.

Figure 2 illustrates an example how the UPP model works. Given an input “so could you tell me”, the optimal sequence of transition actions in Figure 2(b) is calculated based on the UPP model to produce the parsing tree in Figure 2(a). According to the sequence of actions, we can determine the sequence of predicted punctuation symbols like “,NNN?” that have been attached to the words shown in Figure 2(a). The final segmentation with the predicted punctuation insertion could be “so, could you tell me?”.

#### 4.4 Model training and decoding

In practice, the sub-models in Model (6) and (7) with the form of  $P(Y|X)$  is computed with a linear model  $Score(Y, X)$  as

$$Score(Y, X) = \Phi(Y, X) \cdot \vec{\lambda}$$

where  $\Phi(Y, X)$  is the feature vector extracted from the output  $Y$  and the context  $X$ , and  $\vec{\lambda}$  is the weight vector. For the features of the models, we incorporate the bag of words and POS tags as well as tree-based features shown in Table 2, which are the same as those defined in (Zhang and Nivre, 2011).

(a)	$w_s; w_0; w_1; w_2; p_s; p_0; p_1; p_2; w_s p_s; w_0 p_0; w_1 p_1; w_2 p_2; w_s p_s w_0 p_0; w_s p_s w_0; w_s p_s p_0; w_s w_0 p_0; p_s w_0 p_0; w_s w_0; p_s p_0; p_0 p_1; p_s p_0 p_1; p_0 p_1 p_2;$
(b)	$p_{sh} p_s p_0; p_s p_{sl} p_0; p_s p_{sr} p_0; p_s p_0 p_0; w_s d; p_s d; w_0 d; p_0 d; w_s w_0 d; p_s p_0 d; w_s v_l; p_s v_l; w_s v_r; p_s v_r; w_0 v_l; p_0 v_l; w_{sh}; p_{sh}; t_s; w_0 l; p_0 l; t_0 l; w_0 r; p_0 r; t_0 r; w_1 l; p_1 l; t_1 l; w_{sh2}; p_{sh2}; t_{sh}; w_{sl2}; p_{sl2}; t_{sl2}; w_{sr2}; p_{sr2}; t_{sr2}; w_0 l2; p_0 l2; t_0 l2; p_s p_{sl} p_{sl2}; p_s p_{sr} p_{sr2}; p_s p_{sh} p_{sh2}; p_0 p_0 p_0 l2; w_s T_l; p_s T_l; w_s T_r; p_s T_r; w_0 T_l; p_0 T_l;$

Table 2. (a) Features of the bag of words and POS tags. (b). Tree-based features.  $w$ —word;  $p$ —POS tag;  $d$ —distance between  $w_s$  and  $w_0$ ;  $v$ —number of modifiers;  $t$ —dependency label;  $T$ —set of dependency labels;  $s, 0, 1$  and  $2$  index the stack top and three front items in the queue respectively;  $h$ —head;  $l$ —left/leftmost;  $r$ —right/rightmost;  $h2$ —head of a head;  $l2$ —second leftmost;  $r2$ —second rightmost.

The training data for both the CPP and UPP models need to contain parsing trees and punctuation information. Due to the absence of annotation over transcribed speech data, we adapt the Treebank data for the purpose of model training. To do this, we remove all types of syntactic information related to punctuation symbols from the raw Treebank data, but record what punctuation symbols are attached to the words. We normalize various punctuation symbols into two types: Middle-paused punctuation (M) and Full-stop punctuation (F). Plus null type (N), there are three kinds of punctuation symbols attached to the words. Table 3 illustrates the normalizations of punctuation symbols. In the experiments, we did not further distinguish the type among full-stop punctuation because the question mark and the exclamation mark have very low frequency in Treebank data.

But our CPP and UPP models are both independent regarding the number of punctuation types to be predicted.

Punctuations	Normalization
Period, question mark, exclamation mark	Full-stop punctuation (F)
Comma, Colon, semi-colon	Middle-paused punctuation (M)
Multiple Punctuations (e.g., !!!!?)	Full-stop punctuation (F)
Quotations, brackets, etc.	Null (N)

Table 3. Punctuation normalization in training data

As the feature templates are the same for the model training of both CPP and UPP, the training instances of CPP and UPP have the same contexts but with different outputs. Similar to work in (Zhang and Clark, 2008; Zhang and Nivre, 2011), we train CPP and UPP by generalized perceptron (Collins, 2002).

In decoding, beam search is performed to get the optimal sequence of transition actions in CPP and UPP, and the optimal punctuation symbols in CPP. To ensure each segment decided by a full-stop punctuation corresponds to a single parsing tree, two constraints are applied in decoding for the pruning of deficient search paths.

- (1) Proceeding-constraint: If the partial parsing result is not a single tree, the full-stop punctuation prediction in CPP cannot be performed. In UPP, if *Shift(F)* or *RightArc(F)* fail to result in a single parsing tree, they cannot be performed as well.
- (2) Succeeding-constraint: If the full-stop punctuation is predicted in CPP, or *Shift(F)* and *RightArc(F)* are performed in UPP, the following transition actions must be a sequence of *Reduce* actions until the stack becomes empty.

## 5 Experiments

### 5.1 Experimental setup

Our training data of transition-based dependency trees are converted from phrasal structure trees in English Web Treebank (LDC2012T13) and the English portion of OntoNotes 4.0 (LDC2011T03) by the Stanford Conversion toolkit (Marneffe et al., 2006). It contains around 1.5M words in total and consist of various genres including weblogs, web texts, newsgroups, email, reviews, question-

answer sessions, newswires, broadcast news and broadcast conversations. To simulate the transcribed speech text, all words in dependency trees are lowercased and punctuations are excluded before model training. In addition, every ten dependency trees are concatenated sequentially to simulate a parsing result of a stream of words in the model training.

There are two test data sets used in our experiments. One is the English corpus of the IWSLT09 evaluation campaign (Paul, 2009) that is the conversational speech text. The other is a subset of the TDT4 English data (LDC2005T16) which consists of 200 hours of closed-captioned broadcast news.

In the decoding, the beam size of both the transition-based parsing and punctuation prediction is set to 5. The part-of-speech tagger is our re-implementation of the work in (Collins, 2002).

The evaluation metrics of our experiments are precision (*prec.*), recall (*rec.*) and F1-measure ( $F_1$ ).

For the comparison, we also implement a baseline method based on the CRF model. It incorporates the features of bag of words and POS tags shown in Table 2(a), which are commonly used in previous related work.

### 5.2 Experimental results

We test the performance of our method on both the correctly recognized texts and automatically recognized texts. The former data is used to evaluate the capability of punctuation prediction of our algorithm regardless of the noises from speech data, as our model training data come from formal text instead of transcribed speech data. The usage of the latter test data set aims to evaluate the effectiveness of our method in real applications where lots of substantial recognition errors could be contained. In addition, we also evaluate the quality of our transition-based parsing, as its performance could have a big influence on the quality of punctuation prediction.

#### 5.2.1 Performance on correctly recognized text

The evaluation of our method on correctly recognized text uses 10% of IWSLT09 training set, which consists of 19,972 sentences from BTEC (Basic Travel Expression Corpus) and 10,061 sentences from CT (Challenge Task). The average input length is about 10 words and each input contains 1.3 sentences on average. The evaluation results are presented in Table 4.

	Measure	Middle-Paused	Full-stop	Mixed
Baseline (CRF)	<i>prec.</i>	33.2%	81.5%	78.8%
	<i>rec.</i>	25.9%	83.8%	80.7%
	$F_1$	29.1%	82.6%	79.8%
CPP	<i>prec.</i>	51%	89%	89.6%
	<i>rec.</i>	50.3%	93.1%	92.7%
	$F_1$	50.6%	91%	91.1%
UPP	<i>prec.</i>	52.6%	93.2%	92%
	<i>rec.</i>	59.7%	91.3%	92.3%
	$F_1$	55.9%	92.2%	92.2%

Table 4. Punctuation prediction performance on correctly recognized text

We achieved good performance on full-stop punctuation compared to the baseline, which shows our method can efficiently process sentence segmentation because each segment is decided by the structure of a single parsing tree. In addition, the global syntactic knowledge used in our work help capture long range dependencies of punctuations. The performance of middle-paused punctuation prediction is fairly low between all methods, which shows predicting middle-paused punctuations is a difficult task. This is because the usage of middle-paused punctuations is very flexible, especially in conversational data. The last column in Table 4 presents the performance of the pure segmentation task where the middle-paused and full-stop punctuations are mixed and not distinguished. The performance of our method is much higher than that of the baseline, which shows our method is good at segmentation. We also note that UPP yields slightly better performance than CPP on full-stop and mixed punctuation prediction, and much better performance on middle-paused punctuation prediction. This could be because the interaction of parsing and punctuation prediction is closer together in UPP than in CPP.

### 5.2.2 Performance on automatically recognized text

Table 5 shows the experimental results of punctuation prediction on automatically recognized text from TDT4 data that is recognized using SRI’s English broadcast news ASR system where the word error rate is estimated to be 18%. As the annotation of middle-paused punctuations in TDT4 is not available, we can only evaluate the performance of full-stop punctuation prediction (i.e., detecting sentence boundaries). Thus, we merge every three sentences into one single input before

performing full-stop prediction. The average input length is about 43 words.

	Measure	Full-stop
Baseline (CRF)	<i>prec.</i>	37.7%
	<i>rec.</i>	60.7%
	$F_1$	46.5%
CPP	<i>prec.</i>	63%
	<i>rec.</i>	58.6%
	$F_1$	60.2%
UPP	<i>prec.</i>	73.9%
	<i>rec.</i>	51.6%
	$F_1$	60.7%

Table 5. Punctuation prediction performance on automatically recognized text

Generally, the performance shown in Table 5 is not as high as that in Table 4. This is because the speech recognition error from ASR systems degrades the capability of model prediction. Another reason might be that the domain and style of our training data mismatch those of TDT4 data. The baseline gets a little higher recall than our method, which shows the baseline method tends to make aggressive segmentation decisions. However, both precision and  $F_1$  score of our method are much higher than the baseline. CPP has higher recall than UPP, but with lower precision and  $F_1$  score. This is in line with Table 4, which consistently illustrates CPP can get higher recall on full-stop punctuation prediction for both correctly recognized and automatically recognized texts.

### 5.2.3 Performance of transition-based parsing

Performance of parsing affects the quality of punctuation prediction in our work. In this section, we separately evaluate the performance of our transition-based parser over various domains including the Wall Street Journal (WSJ), weblogs, newsgroups, answers, email messages and reviews. We divided annotated Treebank data into three data sets: 90% for model training, 5% for the development set and 5% for the test set. The accuracy of our POS-tagger achieves 96.71%. The beam size in the decoding of both our POS-tagging and parsing is set to 5. Table 6 presents the results of our experiments on the measures of UAS and LAS, where the overall accuracy is obtained from a general model which is trained over the combination of the training data from all domains.

We first evaluate the performance of our transition-based parsing over texts containing punctuations (TCP). The evaluation results show that our transition-based parser achieves state-of-the-art performance levels, referring to the best dependency parsing results reported in the shared task of SANCL 2012 workshop<sup>2</sup>, although they cannot be compared directly due to the different training data and test data sets used in the experiments. Secondly, we evaluate our parsing model in CPP over the texts without punctuations (TOP). Surprisingly, the performance over TOP is better than that over TCP. The reason could be that we cleaned out data noises caused by punctuations when preparing TOP data. These results illustrate that the performance of transition-based parsing in our method does not degrade after being integrated with punctuation prediction. As a by-product of the punctuation prediction task, the outputs of parsing trees can benefit the subsequent text processing tasks.

	Data sets	UAS	LAS
Texts containing punctuations (TCP)	WSJ	92.6%	90.3%
	Weblogs	90.7%	88.2%
	Answers	89.4%	85.7%
	Newsgroups	90.1%	87.6%
	Reviews	90.9%	88.4%
	Email Messages	89.6%	87.1%
	Overall	90.5%	88%
Texts without punctuations (TOP)	WSJ	92.6%	91.1%
	Weblogs	92.5%	91.1%
	Answers	95%	94%
	Newsgroups	92.6%	91.2%
	Reviews	92.6%	91.2%
	Email Messages	92.9%	91.7%
	Overall	92.6%	91.2%

Table 6. The performance of our transition-based parser on written texts. UAS=unlabeled attachment score; LAS=labeled attachment score

## 6 Conclusion and Future Work

In this paper, we proposed a novel method for punctuation prediction of transcribed speech texts. Our approach jointly performs parsing and punctuation prediction by integrating a rich set of syntactic features. It can not only yield parse trees, but also determine sentence boundaries and predict punctuation symbols from a global view of the in-

puts. The proposed algorithm has linear complexity in the size of input, which can efficiently process the stream of words from a purely text processing perspective without the dependences on either the ASR systems or subsequent tasks. The experimental results show that our approach outperforms the CRF-based method on both the correctly recognized and automatically recognized texts. In addition, the performance of the parsing over the stream of transcribed words is state-of-the-art, which can benefit many subsequent text processing tasks.

In future work, we will try our method on other languages such as Chinese and Japanese, where Treebank data is available. We would also like to test the MT performance over transcribed speech texts with punctuation symbols inserted based on our method proposed in this paper.

## References

- B. Bohnet and J. Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In Proc. EMNLP-CoNLL 2012.
- H. Christensen, Y. Gotoh, and S. Renals. 2001. Punctuation annotation using statistical prosody models. In Proc. of ISCA Workshop on Prosody in Speech Recognition and Understanding.
- M. Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In Proc. EMNLP'02, pages 1-8.
- B. Favre, R. Grishman, D. Hillard, H. Ji, D. Hakkani-Tur, and M. Ostendorf. 2008. Punctuating speech for information extraction. In Proc. of ICASSP'08.
- B. Favre, D. HakkaniTur, S. Petrov and D. Klein. 2008. Efficient sentence segmentation using syntactic features. In Spoken Language Technologies (SLT).
- A. Gravano, M. Jansche, and M. Bacchiani. 2009. Restoring punctuation and capitalization in transcribed speech. In Proc. of ICASSP'09.
- J. Hatori, T. Matsuzaki, Y. Miyao and J. Tsujii. 2011. Incremental joint POS tagging and dependency parsing in Chinese. In Proc. Of IJCNLP'11.
- J. Huang and G. Zweig. 2002. Maximum entropy model for punctuation annotation from speech. In Proc. Of ICSLP'02.

<sup>2</sup> <https://sites.google.com/site/sancl2012/home/shared-task/results>

- J.H. Kim and P.C. Woodland. 2001. The use of prosody in a combined system for punctuation generation and speech recognition. In Proc. of EuroSpeech'01.
- Y. Liu, A. Stolcke, E. Shriberg, and M. Harper. 2005. Using conditional random fields for sentence boundary detection in speech. In Proc. of ACL'05.
- W. Lu and H.T. Ng. 2010. Better Punctuation Prediction with Dynamic Conditional Random Fields. In Proc. Of EMNLP'10. Pages 177-186.
- M. Marneffe, B. MacCartney, C.D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In Proc. LREC'06.
- E. Matusov, A. Mauser, and H. Ney. 2006. Automatic sentence segmentation and punctuation prediction for spoken language translation. In Proc. of IWSLT'06.
- S. Nakamura. 2009. Overcoming the language barrier with speech translation technology. In Science & Technology Trends - Quarterly Review. No. 31. April 2009.
- J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In Proceedings of IWPT, pages 149–160, Nancy, France.
- J. Nivre and M. Scholz. 2004. Deterministic dependency parsing of English text. In Proc. COLING'04.
- M. Paul. 2009. Overview of the IWSLT 2009 Evaluation Campaign. In Proceedings of IWSLT'09.
- B. Roark, Y. Liu, M. Harper, R. Stewart, M. Lease, M. Snover, I. Shafran, B. Dorr, J. Hale, A. Krasnyanskaya, and L. Yung. 2006. Reranking for sentence boundary detection in conversational speech. In Proc. ICASSP, 2006.
- A. Stolcke and E. Shriberg, "Automatic linguistic segmentation of conversational speech," Proc. ICSLP, vol. 2, 1996.
- A. Stolcke, E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, G. Tur, and Y. Lu. 1998. Automatic detection of sentence boundaries and disfluencies based on recognized words. In Proc. of ICSLP' 98.
- Takezawa, T. Morimoto, T. Sagisaka, Y. Campbell, N. Iida, H. Sugaya, F. Yokoo, A. Yamamoto, Seiichi. 1998. A Japanese-to-English speech translation system: ATR-MATRIX. In Proc. ICSLP'98.
- Y. Zhang and J. Nivre. 2011. Transition-based Dependency Parsing with Rich Non-local Features. In Proc. of ACL'11, pages 188-193.
- Y. Zhang and S. Clark. A Tale of Two Parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. 2008. In Proc. of EMNLP'08, pages 562-571.

# Discriminative Learning with Natural Annotations: Word Segmentation as a Case Study

Wenbin Jiang<sup>1</sup> Meng Sun<sup>1</sup> Yajuan Lü<sup>1</sup> Yating Yang<sup>2</sup> Qun Liu<sup>3, 1</sup>

<sup>1</sup>Key Laboratory of Intelligent Information Processing  
Institute of Computing Technology, Chinese Academy of Sciences

{jiangwenbin, sunmeng, lvyajuan}@ict.ac.cn

<sup>2</sup>Multilingual Information Technology Research Center

The Xinjiang Technical Institute of Physics & Chemistry, Chinese Academy of Sciences

yangyt@ms.xjb.ac.cn

<sup>3</sup>Centre for Next Generation Localisation

Faculty of Engineering and Computing, Dublin City University

qliu@computing.dcu.ie

## Abstract

Structural information in web text provides natural annotations for NLP problems such as word segmentation and parsing. In this paper we propose a discriminative learning algorithm to take advantage of the linguistic knowledge in large amounts of natural annotations on the Internet. It utilizes the Internet as an external corpus with massive (although slight and sparse) natural annotations, and enables a classifier to evolve on the large-scaled and real-time updated web text. With Chinese word segmentation as a case study, experiments show that the segmenter enhanced with the Chinese wikipedia achieves significant improvement on a series of testing sets from different domains, even with a single classifier and local features.

## 1 Introduction

Problems related to information retrieval, machine translation and social computing need fast and accurate text processing, for example, word segmentation and parsing. Taking Chinese word segmentation for example, the state-of-the-art models (Xue and Shen, 2003; Ng and Low, 2004; Gao et al., 2005; Nakagawa and Uchimoto, 2007; Zhao and Kit, 2008; Jiang et al., 2009; Zhang and Clark, 2010; Sun, 2011b; Li, 2011) are usually trained on human-annotated corpora such as the Penn Chinese Treebank (CTB) (Xue et al., 2005), and perform quite well on corresponding test sets. Since the text used for corpus annotating are usually drawn from specific fields (e.g. newswire or finance), and the annotated corpora are limited in

“... think that *NLP* has already ...”

... 认为 自然语言处理 已经 ...  
i-1 i j j+1

(a) Natural annotation by hyperlink

... 认为 || 自然语言处理 || 已经 ...  
i-1 i j j+1

(b) Knowledge for word segmentation

... 认为 自然语言处理 已经 ...  
i-1 i j j+1

(c) Knowledge for dependency parsing

Figure 1: Natural annotations for word segmentation and dependency parsing.

size (e.g. tens of thousands), the performance of word segmentation tends to degrade sharply when applied to new domains.

Internet provides large amounts of raw text, and statistics collected from it have been used to improve parsing performance (Nakov and Hearst, 2005; Pitler et al., 2010; Bansal and Klein, 2011; Zhou et al., 2011). The Internet also gives massive (although slight and sparse) natural annotations in the forms of structural information including hyperlinks, fonts, colors and layouts (Sun, 2011a). These annotations usually imply valuable knowledge for problems such as word segmentation and parsing, based on the hypothesis that the subsequences marked by structural information are meaningful fragments in sentences. Figure 1 shows an example. The hyperlink indicates

a Chinese phrase (meaning NLP), and it probably corresponds to a connected sub-graph for dependency parsing. Creators of web text give valuable annotations during editing, the whole Internet can be treated as a wide-covered and real-time updated corpus.

Different from the dense and accurate annotations in human-annotated corpora, natural annotations in web text are sparse and slight, it makes direct training of NLP models impracticable. In this work we take for example a most important problem, word segmentation, and propose a novel discriminative learning algorithm to leverage the knowledge in massive natural annotations of web text. Character classification models for word segmentation usually factorize the whole prediction into atomic predictions on characters (Xue and Shen, 2003; Ng and Low, 2004). Natural annotations in web text can be used to get rid of implausible predication candidates for related characters, knowledge in the natural annotations is therefore introduced in the manner of searching space pruning. Since constraint decoding in the pruned searching space integrates the knowledge of the baseline model and natural annotations, it gives predictions not worse than the normal decoding does. Annotation differences between the outputs of constraint decoding and normal decoding are used to train the enhanced classifier. This strategy makes the usage of natural annotations simple and universal, which facilitates the utilization of massive web text and the extension to other NLP problems.

Although there are lots of choices, we choose the Chinese wikipedia as the knowledge source due to its high quality. Structural information, including hyperlinks, fonts and colors are used to determine the boundaries of meaningful fragments. Experimental results show that, the knowledge implied in the natural annotations can significantly improve the performance of a baseline segmenter trained on CTB 5.0, an F-measure increment of 0.93 points on CTB test set, and an average increment of 1.53 points on 7 other domains. It is an effective and inexpensive strategy to build word segmenters adaptive to different domains. We hope to extend this strategy to other NLP problems such as named entity recognition and parsing.

In the rest of the paper, we first briefly introduce the problems of Chinese word segmentation and the character classification model in section

Type	Templates	Instances
$n$ -gram	$C_{-2}$	$C_{-2}$ =认
	$C_{-1}$	$C_{-1}$ =为
	$C_0$	$C_0$ =自
	$C_1$	$C_1$ =然
	$C_2$	$C_2$ =语
	$C_{-2}C_{-1}$	$C_{-2}C_{-1}$ =认为
	$C_{-1}C_0$	$C_{-1}C_0$ =为自
	$C_0C_1$	$C_0C_1$ =自然
	$C_1C_2$	$C_1C_2$ =然语
	$C_{-1}C_1$	$C_{-1}C_1$ =为然
function	$Pu(C_0)$	$Pu(C_0)$ =false
	$T(C_{-2:2})$	$T(C_{-2:2})$ =44444

Table 1: Feature templates and instances for character classification-based word segmentation model. Suppose we are considering the  $i$ -th character “自” in “...认为自 然语言处理已经...”.

2, then describe the representation of the knowledge in natural annotations of web text in section 3, and finally detail the strategy of discriminative learning on natural annotations in section 4. After giving the experimental results and analysis in section 5, we briefly introduce the previous related work and then give the conclusion and the expectation of future research.

## 2 Character Classification Model

Character classification models for word segmentation factorize the whole prediction into atomic predictions on single characters (Xue and Shen, 2003; Ng and Low, 2004). Although natural annotations in web text do not directly support the discriminative training of segmentation models, they do get rid of the implausible candidates for predictions of related characters.

Given a sentence as a sequence of  $n$  characters, word segmentation splits the sequence into  $m(\leq n)$  subsequences, each of which indicates a meaningful word. Word segmentation can be formalized as a character classification problem (Xue and Shen, 2003), where each character in the sentence is given a boundary tag representing its position in a word. We adopt the boundary tags of Ng and Low (2004),  $b$ ,  $m$ ,  $e$  and  $s$ , where  $b$ ,  $m$  and  $e$  mean the beginning, the middle and the end of a word, and  $s$  indicates a single-character word. the decoding procedure searches for the labeled character sequence  $y$  that maximizes the score func-

---

**Algorithm 1** Perceptron training algorithm.

---

- 1: **Input:** Training corpus  $\mathcal{C}$
  - 2:  $\vec{\alpha} \leftarrow \mathbf{0}$
  - 3: **for**  $t \leftarrow 1 \dots T$  **do**  $\triangleright T$  iterations
  - 4:   **for**  $(x, \tilde{y}) \in \mathcal{C}$  **do**
  - 5:      $y \leftarrow \arg \max_y \Phi(x, y) \cdot \vec{\alpha}$
  - 6:     **if**  $y \neq \tilde{y}$  **then**
  - 7:        $\vec{\alpha} \leftarrow \vec{\alpha} + \Phi(x, \tilde{y}) - \Phi(x, y)$
  - 8: **Output:** Parameters  $\vec{\alpha}$
- 

tion:

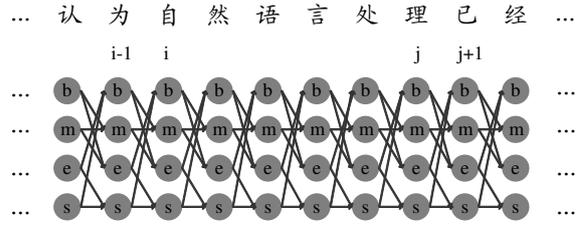
$$\begin{aligned} f(x) &= \arg \max_y S(y | \vec{\alpha}, \Phi, x) \\ &= \arg \max_y \Phi(x, y) \cdot \vec{\alpha} \\ &= \arg \max_y \sum_{(i,t) \in y} \Phi(i, t, x, y) \cdot \vec{\alpha} \end{aligned} \quad (1)$$

The score of the whole sequence  $y$  is accumulated across all its character-label pairs,  $(i, t) \in y$  (s.t.  $1 \leq i \leq n$  and  $t \in \{b, m, e, s\}$ ). The feature function  $\Phi$  maps a labeled sequence or a character-label pair into a feature vector,  $\vec{\alpha}$  is the parameter vector and  $\Phi(x, y) \cdot \vec{\alpha}$  is the inner product of  $\Phi(x, y)$  and  $\vec{\alpha}$ .

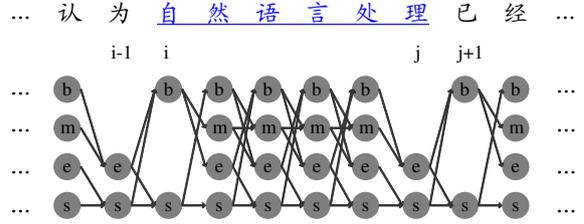
Analogous to other sequence labeling problems, word segmentation can be solved through a viterbi-style decoding procedure. We omit the decoding algorithm in this paper due to its simplicity and popularity.

The feature templates for the classifier is shown in Table 1.  $C_0$  denotes the current character, while  $C_{-k}/C_k$  denote the  $k$ th character to the left/right of  $C_0$ . The function  $Pu(\cdot)$  returns *true* for a punctuation character and *false* for others, the function  $T(\cdot)$  classifies a character into four types, 1, 2, 3 and 4, representing *number*, *date*, *English letter* and *others*, respectively.

The classifier can be trained with online learning algorithms such as perceptron, or offline learning models such as support vector machines. We choose the perceptron algorithm (Collins, 2002) to train the classifier for the character classification-based word segmentation model. It learns a discriminative model mapping from the inputs  $x \in X$  to the outputs  $\tilde{y} \in Y$ , where  $X$  is the set of sentences in the training corpus and  $Y$  is the set of corresponding labeled results. Algorithm 1 shows the perceptron algorithm for tuning the parameter  $\vec{\alpha}$ . The ‘‘averaged parameters’’ technology (Collins, 2002) is used for better performance.



(a) Original searching space



(b) Shrunk searching space

Figure 2: Shrink of searching space for the character classification-based word segmentation model.

### 3 Knowledge in Natural Annotations

Web text gives massive natural annotations in the form of structural informations, including hyperlinks, fonts, colors and layouts (Sun, 2011a). Although slight and sparse, these annotations imply valuable knowledge for problems such as word segmentation and parsing.

As shown in Figure 1, the subsequence  $P = i..j$  of sentence  $S$  is composed of bolded characters determined by a hyperlink. Such natural annotations do not clearly give each character a boundary tag, or define the head-modifier relationship between two words. However, they do help to shrink the set of plausible predication candidates for each character or word. For word segmentation, it implies that characters  $i - 1$  and  $j$  are the rightmost characters of words, while characters  $i$  and  $j + 1$  are the leftmost characters of words. For  $i - 1$  or  $j$ , the plausible predication set  $\Psi$  becomes  $\{e, s\}$ ; For  $i$  and  $j + 1$ , it becomes  $\{b, s\}$ ; For other characters  $c$  except the two at sentence boundaries,  $\Psi(c)$  is still  $\{b, m, e, s\}$ . For dependency parsing, the subsequence  $P$  tends to form a connected dependency graph if it contains more than one word. Here we use  $\Psi$  to denote the set of plausible head of a word (modifier). There must be a single word  $w \in P$  as the root of subsequence  $P$ , whose plausible heads fall out of  $P$ , that is,  $\Psi(w) = \{x | x \in S - P\}$ . For the words in  $P$  except the root, the plausible heads for each

---

**Algorithm 2** Perceptron learning with natural annotations.

---

```
1:  $\vec{\alpha} \leftarrow \text{TRAIN}(\mathcal{C})$ 
2: for  $x \in \mathcal{F}$  do
3:    $y \leftarrow \text{DECODE}(x, \vec{\alpha})$ 
4:    $\tilde{y} \leftarrow \text{CONSTRAINTDECODE}(x, \vec{\alpha}, \Psi)$ 
5:   if  $y \neq \tilde{y}$  then
6:      $\mathcal{C}' \leftarrow \mathcal{C}' \cup \{\tilde{y}\}$ 
7:  $\vec{\alpha} \leftarrow \text{TRAIN}(\mathcal{C} \cup \mathcal{C}')$ 
```

---

word  $w$  are the words in  $P$  except  $w$  itself, that is,  $\Psi(w) = \{x | x \in P - \{w\}\}$ .

Creators of web text give valuable structural annotations during editing, these annotations reduce the predication uncertainty for atomic characters or words, although not exactly defining which predication is. Figure 2 shows an example for word segmentation, depicting the shrink of searching space for the character classification-based model. Since the decrement of uncertainty indicates the increment of knowledge, the whole Internet can be treated as a wide-covered and real-time updated corpus. We choose the Chinese wikipedia as the external knowledge source, and structural information including hyperlinks, fonts and colors are used in the current work due to their explicitness of representation.

#### 4 Learning with Natural Annotations

Different from the dense and accurate annotations in human-annotated corpora, natural annotations are sparse and slight, which makes direct training of NLP models impracticable. Annotations implied by structural information do not give an exact predication to a character, however, they help to get rid of the implausible predication candidates for related characters, as described in the previous section.

Previous work on constituency parsing or machine translation usually resort to some kinds of heuristic tricks, such as punctuation restrictions, to eliminate some implausible candidates during decoding. Here the natural annotations also bring knowledge in the manner of searching space pruning. Conditioned on the completeness of the decoding algorithm, a model trained on an existing corpus probably gives better or at least not worse predications, by constraint decoding in the pruned searching space. The constraint decoding procedure integrates the knowledge of the baseline

---

**Algorithm 3** Online version of perceptron learning with natural annotations.

---

```
1:  $\vec{\alpha} \leftarrow \text{TRAIN}(\mathcal{C})$ 
2: for  $x$  with natural annotations do
3:    $y \leftarrow \text{DECODE}(x, \vec{\alpha})$ 
4:    $\tilde{y} \leftarrow \text{CONSTRAINTDECODE}(x, \vec{\alpha}, \Psi)$ 
5:   if  $y \neq \tilde{y}$  then
6:      $\vec{\alpha} \leftarrow \vec{\alpha} + \Phi(x, \tilde{y}) - \Phi(x, y)$ 
7:   output  $\vec{\alpha}$  at regular time
```

---

model and natural annotations, the predication differences between the outputs of constraint decoding and normal decoding can be used to train the enhanced classifier.

Restrictions of the searching space according to natural annotations can be easily incorporated into the decoder. If the completeness of the searching algorithm can be guaranteed, the constraint decoding in the pruned searching space will give predications not worse than those given by the normal decoding. If a predication of constraint decoding differs from that of normal decoding, it indicates that the annotation precision is higher than the latter. Furthermore, the degree of difference between the two predications represents the amount of new knowledge introduced by the natural annotations over the baseline.

The baseline model  $\vec{\alpha}$  is trained on an existing human-annotated corpus. A set of sentences  $\mathcal{F}$  with natural annotations are extracted from the Chinese wikipedia, and we reserve the ones for which constraint decoding and normal decoding give different predications. The predictions of reserved sentences by constraint decoding are used as additional training data for the enhanced classifier. The overall training pipeline is analogous to self-training (McClosky et al., 2006), Algorithm 2 shows the pseudo-codes. Considering the *online* characteristic of the perceptron algorithm, if we are able to leverage much more (than the Chinese wikipedia) data with natural annotations, an online version of learning procedure shown in Algorithm 3 would be a better choice. The technology of “averaged parameters” (Collins, 2002) is easily to be adapted here for better performance.

When constraint decoding and normal decoding give different predications, we only know that the former is probably better than the latter. Although there is no explicit evidence for us to measure how much difference in accuracy between the

Partition	Sections	# of word
<b>CTB</b>		
Training	1 – 270	0.47M
	400 – 931	
	1001 – 1151	
Developing	301 – 325	6.66K
Testing	271 – 300	7.82K

Table 2: Data partitioning for CTB 5.0.

two predications, we can approximate how much new knowledge that a naturally annotated sentence brings. For a sentence  $x$ , given the predications of constraint decoding and normal decoding,  $\tilde{y}$  and  $y$ , the difference of their scores  $\delta = S(y) - S(\tilde{y})$  indicates the degree to which the current model mistakes. This indicator helps us to select more valuable training examples.

The strategy of learning with natural annotations can be adapted to other situations. For example, if we have a list of words or phrases (especially in a specific domain such as medicine and chemical), we can generate annotated sentences automatically by string matching in a large amount of raw text. It probably provides a simple and effective domain adaptation strategy for already trained models.

## 5 Experiments

We use the Penn Chinese Treebank 5.0 (CTB) (Xue et al., 2005) as the existing annotated corpus for Chinese word segmentation. For convenient of comparison with other work in word segmentation, the whole corpus is split into three partitions as follows: chapters 271-300 for testing, chapters 301-325 for developing, and others for training. We choose the Chinese wikipedia<sup>1</sup> (version 20120812) as the external knowledge source, because it has high quality in contents and it is much better than usual web text. Structural informations, including hyperlinks, fonts and colors are used to derive the annotation information.

To further evaluate the improvement brought by the fuzzy knowledge in Chinese wikipedia, a series of testing sets from different domains are adopted. The four testing sets from SIGHAN Bakeoff 2010 (Zhao and Liu, 2010) are used, they are drawn from the domains of literature, finance, computer science and medicine. Although the reference sets are annotated according to a different

<sup>1</sup><http://download.wikimedia.org/backup-index.html>.

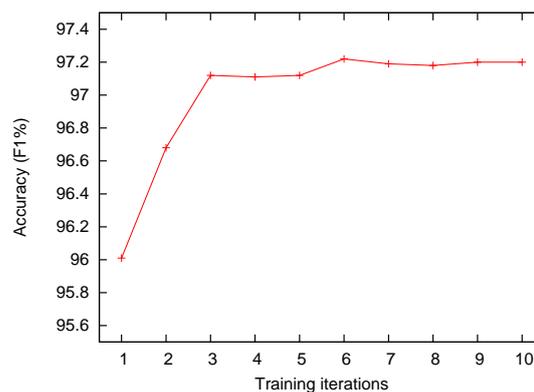


Figure 3: Learning curve of the averaged perceptron classifier on the CTB developing set.

word segmentation standard (Yu et al., 2001), the quantity of accuracy improvement is still illustrative since there are no vast diversities between the two segmentation standards. We also annotated another three testing sets<sup>2</sup>, their texts are drawn from the domains of chemistry, physics and machinery, and each contains 500 sentences.

### 5.1 Baseline Classifier for Word Segmentation

We train the baseline perceptron classifier for word segmentation on the training set of CTB 5.0, using the developing set to determine the best training iterations. The performance measurement for word segmentation is balanced F-measure,  $F = 2PR/(P + R)$ , a function of precision  $P$  and recall  $R$ , where  $P$  is the percentage of words in segmentation results that are segmented correctly, and  $R$  is the percentage of correctly segmented words in the gold standard words.

Figure 3 shows the learning curve of the averaged perceptron on the developing set. The second column of Table 3 lists the performance of the baseline classifier on eight testing sets, where newswire denotes the testing set of the CTB itself. The classifier performs much worse on the domains of chemistry, physics and machinery, it indicates the importance of domain adaptation for word segmentation (Gao et al., 2004; Ma and Way, 2009; Gao et al., 2010). The accuracy on the testing sets from SIGHAN Bakeoff 2010 is even lower due to the difference in both domains and word segmentation standards.

<sup>2</sup>They are available at <http://nlp.ict.ac.cn/jiangwenbin/>.

Dataset	Baseline (F%)	Enhanced (F%)	
NewsWire	97.35	98.28	+0.93
<b>Out-of-Domain</b>			
Chemistry	93.61	95.68	+2.07
Physics	95.10	97.24	+2.14
Machinery	96.08	97.66	+1.58
Literature	92.42	93.53	+1.11
Finance	92.50	93.16	+0.66
Computer	89.46	91.19	+1.73
Medicine	91.88	93.34	+1.46
<b>Average</b>	<b>93.01</b>	<b>94.54</b>	<b>+1.53</b>

Table 3: Performance of the baseline classifier and the classifier enhanced with natural annotations in Chinese wikipedia.

## 5.2 Classifier Enhanced with Natural Annotations

The Chinese wikipedia contains about 0.5 million items. From their description text, about 3.9 millions of sentences with natural annotations are extracted. With the CTB training set as the existing corpus  $\mathcal{C}$ , about 0.8 million sentences are reserved according to Algorithm 2, the segmentations given by constraint decoding are used as additional training data for the enhanced classifier.

According to the previous description, the difference of the scores of constraint decoding and normal decoding,  $\delta = S(y) - S(\tilde{y})$ , indicates the importance of a constraint segmentation to the improvement of the baseline classifier. The constraint segmentations of the reserved sentences are sorted in descending order according to the difference of the scores of constraint decoding and normal decoding, as described previously. From the beginning of the sorted list, different amounts of segmented sentences are used as the additional training data for the enhanced character classifier. Figure 4 shows the performance curve of the enhanced classifiers on the developing set of CTB. We found that the highest accuracy was achieved when 160,000 sentences were used, while more additional training data did not give continuous improvement. A recent related work about self-training for segmentation (Liu and Zhang, 2012) also reported a very similar trend, that only a moderate amount of raw data gave the most obvious improvements.

The performance of the enhanced classifier is listed in the third column of Table 3. On the CTB testing set, training data from the Chinese

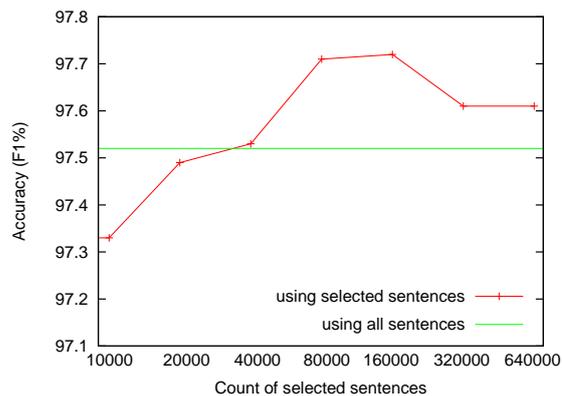


Figure 4: Performance curve of the classifier enhanced with selected sentences of different scales.

Model	Accuracy (F%)
(Jiang et al., 2008)	97.85
(Kruengkrai et al., 2009)	97.87
(Zhang and Clark, 2010)	97.79
(Wang et al., 2011)	98.11
(Sun, 2011b)	98.17
<b>Our Work</b>	<b>98.28</b>

Table 4: Comparison with state-of-the-art work in Chinese word segmentation.

wikipedia brings an F-measure increment of 0.93 points. On out-of-domain testing sets, the improvements are much larger, an average increment of 1.53 points is achieved on seven domains. It is probably because the distribution of the knowledge in the CTB training data is concentrated in the domain of newswire, while the contents of the Chinese wikipedia cover a broad range of domains, it provides knowledge complementary to that of CTB.

Table 4 shows the comparison with other work in Chinese word segmentation. Our model achieves an accuracy higher than that of the state-of-the-art models trained on CTB only, although using a single classifier with only local features. From the viewpoint of resource utilization, the comparison between our system and previous work without using additional training data is unfair. However, we believe this work shows another interesting way to improve Chinese word segmentation, it focuses on the utilization of fuzzy and sparse knowledge on the Internet rather than making full use of a specific human-annotated corpus. On the other hand, since only a single classifier and local features are used in our method, better performance could be achieved

resorting to complicated features, system combination and other semi-supervised technologies. What is more, since the text on Internet is wide-covered and real-time updated, our strategy also helps a word segmenter be more domain adaptive and up to date.

## 6 Related Work

Li and Sun (2009) extracted character classification instances from raw text for Chinese word segmentation, resorting to the indication of punctuation marks between characters. Sun and Xu (Sun and Xu, 2011) utilized the features derived from large-scaled unlabeled text to improve Chinese word segmentation. Although the two work also made use of large-scaled raw text, our method is essentially different from theirs in the aspects of both the source of knowledge and the learning strategy.

Lots of efforts have been devoted to semi-supervised methods in sequence labeling and word segmentation (Xu et al., 2008; Suzuki and Isozaki, 2008; Haffari and Sarkar, 2008; Tomanek and Hahn, 2009; Wang et al., 2011). A semi-supervised method tries to find an optimal hyper-plane of both annotated data and raw data, thus to result in a model with better coverage and higher accuracy. Researchers have also investigated unsupervised methods in word segmentation (Zhao and Kit, 2008; Johnson and Goldwater, 2009; Mochihashi et al., 2009; Hewlett and Cohen, 2011). An unsupervised method mines the latent distribution regularity in the raw text, and automatically induces word segmentation knowledge from it. Our method also needs large amounts of external data, but it aims to leverage the knowledge in the fuzzy and sparse annotations. It is fundamentally different from semi-supervised and unsupervised methods in that we aimed to excavate a totally different kind of knowledge, the natural annotations implied by the structural information in web text.

In recent years, much work has been devoted to the improvement of word segmentation in a variety of ways. Typical approaches include the introduction of global training or complicated features (Zhang and Clark, 2007; Zhang and Clark, 2010), the investigation of word internal structures (Zhao, 2009; Li, 2011), the adjustment or adaptation of word segmentation standards (Wu, 2003; Gao et al., 2004; Jiang et al., 2009), the integrated

solution of segmentation and related tasks such as part-of-speech tagging and parsing (Zhou and Su, 2003; Zhang et al., 2003; Fung et al., 2004; Goldberg and Tsarfaty, 2008), and the strategies of hybrid or stacked modeling (Nakagawa and Uchimoto, 2007; Kruengkrai et al., 2009; Wang et al., 2010; Sun, 2011b).

In parsing, Pereira and Schabes (1992) proposed an extended inside-outside algorithm that infers the parameters of a stochastic CFG from a partially parsed treebank. It uses partial bracketing information to improve parsing performance, but it is specific to constituency parsing, and its computational complexity makes it impractical for massive natural annotations in web text. There are also work making use of word co-occurrence statistics collected in raw text or Internet n-grams to improve parsing performance (Nakov and Hearst, 2005; Pitler et al., 2010; Zhou et al., 2011; Bansal and Klein, 2011). When enriching the related work during writing, we found a work on dependency parsing (Spitkovsky et al., 2010) who utilized parsing constraints derived from hypertext annotations to improve the unsupervised dependency grammar induction. Compared with their method, the strategy we proposed is formal and universal, the discriminative learning strategy and the quantitative measurement of fuzzy knowledge enable more effective utilization of the natural annotation on the Internet when adapted to parsing.

## 7 Conclusion and Future Work

This work presents a novel discriminative learning algorithm to utilize the knowledge in the massive natural annotations on the Internet. Natural annotations implied by structural information are used to decrease the searching space of the classifier, then the constraint decoding in the pruned searching space gives predictions not worse than the normal decoding does. Annotation differences between the outputs of constraint decoding and normal decoding are used to train the enhanced classifier, linguistic knowledge in the human-annotated corpus and the natural annotations of web text are thus integrated together. Experiments on Chinese word segmentation show that, the enhanced word segmenter achieves significant improvement on testing sets of different domains, although using a single classifier with only local features.

Since the contents of web text cover a broad range of domains, it provides knowledge comple-

mentary to that of human-annotated corpora with concentrated distribution of domains. The content on the Internet is large-scaled and real-time updated, it compensates for the drawback of expensive building and updating of corpora. Our strategy, therefore, enables us to build a classifier more domain adaptive and up to date. In the future, we will compare this method with self-training to better illustrate the importance of boundary information, and give error analysis on what types of errors are reduced by the method to make this investigation more complete. We will also investigate more efficient algorithms to leverage more massive web text with natural annotations, and further extend the strategy to other NLP problems such as named entity recognition and parsing.

### Acknowledgments

The authors were supported by National Natural Science Foundation of China (Contracts 61202216), 863 State Key Project (No. 2011AA01A207), and National Key Technology R&D Program (No. 2012BAH39B03). Qun Liu's work was partially supported by Science Foundation Ireland (Grant No.07/CE/I1142) as part of the CNGL at Dublin City University. Sincere thanks to the three anonymous reviewers for their thorough reviewing and valuable suggestions!

### References

- Mohit Bansal and Dan Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of ACL*.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*, pages 1–8, Philadelphia, USA.
- Pascale Fung, Grace Ngai, Yongsheng Yang, and Benfeng Chen. 2004. A maximum-entropy chinese parser augmented by transformation-based learning. In *Proceedings of TALIP*.
- Jianfeng Gao, Andi Wu, Mu Li, Chang-Ning Huang, Hongqiao Li, Xinsong Xia, and Haowei Qin. 2004. Adaptive chinese word segmentation. In *Proceedings of ACL*.
- Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. 2005. Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics*.
- Wenjun Gao, Xipeng Qiu, and Xuanjing Huang. 2010. Adaptive chinese word segmentation with online passive-aggressive algorithm. In *Proceedings of CIPS-SIGHAN Workshop*.
- Yoav Goldberg and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *Proceedings of ACL-HLT*.
- Gholamreza Haffari and Anoop Sarkar. 2008. Homotopy-based semi-supervised hidden markov models for sequence labeling. In *Proceedings of COLING*.
- Daniel Hewlett and Paul Cohen. 2011. Fully unsupervised word segmentation with bve and mdl. In *Proceedings of ACL*.
- Wenbin Jiang, Liang Huang, Yajuan Lv, and Qun Liu. 2008. A cascaded linear model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL*.
- Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic adaptation of annotation standards: Chinese word segmentation and pos tagging—a case study. In *Proceedings of the 47th ACL*.
- Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of NAACL*.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *Proceedings of ACL-IJCNLP*.
- Zhongguo Li and Maosong Sun. 2009. Punctuation as implicit annotations for chinese word segmentation. *Computational Linguistics*.
- Zhongguo Li. 2011. Parsing the internal structure of words: A new paradigm for chinese word segmentation. In *Proceedings of ACL*.
- Yang Liu and Yue Zhang. 2012. Unsupervised domain adaptation for joint segmentation and pos-tagging. In *Proceedings of COLING*.
- Yanjun Ma and Andy Way. 2009. Bilingually motivated domain-adapted word segmentation for statistical machine translation. In *Proceedings of EACL*.
- David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the HLT-NAACL*.
- Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested pitman-yor language modeling. In *Proceedings of ACL-IJCNLP*.
- Tetsuji Nakagawa and Kiyotaka Uchimoto. 2007. A hybrid approach to word segmentation and pos tagging. In *Proceedings of ACL*.
- Preslav Nakov and Marti Hearst. 2005. Using the web as an implicit training set: Application to structural ambiguity resolution. In *Proceedings of HLT-EMNLP*.

- Hwee Tou Ng and Jin Kiat Low. 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In *Proceedings of EMNLP*.
- Fernando Pereira and Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of ACL*.
- Emily Pitler, Shane Bergsma, Dekang Lin, and Kenneth Church. 2010. Using web-scale n-grams to improve base np parsing performance. In *Proceedings of COLING*.
- Valentin I. Spitskovsky, Daniel Jurafsky, and Hiyan Alshawi. 2010. Profiting from mark-up: Hyper-text annotations for guided parsing. In *Proceedings of ACL*.
- Weiwei Sun and Jia Xu. 2011. Enhancing chinese word segmentation using unlabeled data. In *Proceedings of EMNLP*.
- Maosong Sun. 2011a. Natural language processing based on naturally annotated web resources. *CHINESE INFORMATION PROCESSING*.
- Weiwei Sun. 2011b. A stacked sub-word model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL*.
- Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using gigaword scale unlabeled data. In *Proceedings of ACL*.
- Katrin Tomanek and Udo Hahn. 2009. Semi-supervised active learning for sequence labeling. In *Proceedings of ACL*.
- Kun Wang, Chengqing Zong, and Keh-Yih Su. 2010. A character-based joint model for chinese word segmentation. In *Proceedings of COLING*.
- Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving chinese word segmentation and pos tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of IJCNLP*.
- Andi Wu. 2003. Customizable segmentation of morphologically derived words in chinese. *Computational Linguistics and Chinese Language Processing*.
- Jia Xu, Jianfeng Gao, Kristina Toutanova, and Hermann Ney. 2008. Bayesian semi-supervised chinese word segmentation for statistical machine translation. In *Proceedings of COLING*.
- Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as lmr tagging. In *Proceedings of SIGHAN Workshop*.
- Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*.
- Shiwen Yu, Jianming Lu, Xuefeng Zhu, Huiming Duan, Shiyong Kang, Honglin Sun, Hui Wang, Qiang Zhao, and Weidong Zhan. 2001. Processing norms of modern chinese corpus. Technical report.
- Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of ACL 2007*.
- Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and pos-tagging using a single discriminative model. In *Proceedings of EMNLP*.
- Huaping Zhang, Hongkui Yu, Deyi Xiong, and Qun Liu. 2003. Hhmm-based chinese lexical analyzer icclas. In *Proceedings of SIGHAN Workshop*.
- Hai Zhao and Chunyu Kit. 2008. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *Proceedings of SIGHAN Workshop*.
- Hongmei Zhao and Qun Liu. 2010. The cips-sighan clp 2010 chinese word segmentation bakeoff. In *Proceedings of CIPS-SIGHAN Workshop*.
- Hai Zhao. 2009. Character-level dependencies in chinese: Usefulness and learning. In *Proceedings of EACL*.
- Guodong Zhou and Jian Su. 2003. A chinese efficient analyser integrating word segmentation, part-of-speech tagging, partial parsing and full parsing. In *Proceedings of SIGHAN Workshop*.
- Guangyou Zhou, Jun Zhao, Kang Liu, and Li Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of ACL*.

# Graph-based Semi-Supervised Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging

Xiaodong Zeng<sup>†</sup> Derek F. Wong<sup>†</sup> Lidia S. Chao<sup>†</sup> Isabel Trancoso<sup>‡</sup>

<sup>†</sup>Department of Computer and Information Science, University of Macau

<sup>‡</sup>INESC-ID / Instituto Superior Técnico, Lisboa, Portugal

nlp2ct.samuel@gmail.com, {derekfw, lidiasc}@umac.mo,  
isabel.trancoso@inesc-id.pt

## Abstract

This paper introduces a graph-based semi-supervised joint model of Chinese word segmentation and part-of-speech tagging. The proposed approach is based on a graph-based label propagation technique. One constructs a nearest-neighbor similarity graph over all trigrams of labeled and unlabeled data for propagating syntactic information, i.e., label distributions. The derived label distributions are regarded as virtual evidences to regularize the learning of linear conditional random fields (CRFs) on unlabeled data. An inductive character-based joint model is obtained eventually. Empirical results on Chinese tree bank (CTB-7) and Microsoft Research corpora (MSR) reveal that the proposed model can yield better results than the supervised baselines and other competitive semi-supervised CRFs in this task.

## 1 Introduction

Word segmentation and part-of-speech (POS) tagging are two critical and necessary initial procedures with respect to the majority of high-level Chinese language processing tasks such as syntax parsing, information extraction and machine translation. The traditional way of segmentation and tagging is performed in a pipeline approach, first segmenting a sentence into words, and then assigning each word a POS tag. The pipeline approach is very simple to implement, but frequently causes error propagation, given that wrong segmentations in the earlier stage harm the subsequent POS tagging (Ng and Low, 2004). The joint approaches of word segmentation and POS tagging (joint S&T) are proposed to resolve these two tasks simultaneously. They effectively alleviate the error propagation, because segmentation

and tagging have strong interaction, given that most segmentation ambiguities cannot be resolved without considering the surrounding grammatical constructions encoded in a POS sequence (Qian and Liu, 2012).

In the past years, several proposed supervised joint models (Ng and Low, 2004; Zhang and Clark, 2008; Jiang et al., 2009; Zhang and Clark, 2010) achieved reasonably accurate results, but the outstanding problem among these models is that they rely heavily on a large amount of labeled data, i.e., segmented texts with POS tags. However, the production of such labeled data is extremely time-consuming and expensive (Jiao et al., 2006; Jiang et al., 2009). Therefore, semi-supervised joint S&T appears to be a natural solution for easily incorporating accessible unlabeled data to improve the joint S&T model. This study focuses on using a graph-based label propagation method to build a semi-supervised joint S&T model. Graph-based label propagation methods have recently shown they can outperform the state-of-the-art in several natural language processing (NLP) tasks, e.g., POS tagging (Subramanya et al., 2010), knowledge acquisition (Talukdar et al., 2008), shallow semantic parsing for unknown predicate (Das and Smith, 2011). As far as we know, however, these methods have not yet been applied to resolve the problem of joint Chinese word segmentation (CWS) and POS tagging.

Motivated by the works in (Subramanya et al., 2010; Das and Smith, 2011), for structured problems, graph-based label propagation can be employed to infer valuable syntactic information (n-gram-level label distributions) from labeled data to unlabeled data. This study extends this intuition to construct a similarity graph for propagating trigram-level label distributions. The derived label distributions are regarded as prior knowledge to regularize the learning of a sequential model, conditional random fields (CRFs) in this case, on both

labeled and unlabeled data to achieve the semi-supervised learning. The approach performs the incorporation of the derived labeled distributions by manipulating a “virtual evidence” function as described in (Li, 2009). Experiments on the data from the Chinese tree bank (CTB-7) and Microsoft Research (MSR) show that the proposed model results in significant improvement over other comparative candidates in terms of F-score and out-of-vocabulary (OOV) recall.

This paper is structured as follows: Section 2 points out the main differences with the related work of this study. Section 3 reviews the background, including supervised character-based joint S&T model based on CRFs and graph-based label propagation. Section 4 presents the details of the proposed approach. Section 5 reports the experiment results. The conclusion is drawn in Section 6.

## 2 Related Work

Prior supervised joint S&T models present approximate 0.2% - 1.3% improvement in F-score over supervised pipeline ones. The state-of-the-art joint models include reranking approaches (Shi and Wang, 2007), hybrid approaches (Nakagawa and Uchimoto, 2007; Jiang et al., 2008; Sun, 2011), and single-model approaches (Ng and Low, 2004; Zhang and Clark, 2008; Kruengkrai et al., 2009; Zhang and Clark, 2010). The proposed approach in this paper belongs to the single-model type.

There are few explorations of semi-supervised approaches for CWS or POS tagging in previous works. Xu et al. (2008) described a Bayesian semi-supervised CWS model by considering the segmentation as the hidden variable in machine translation. Unlike this model, the proposed approach is targeted at a general model, instead of one oriented to machine translation task. Sun and Xu (2011) enhanced a CWS model by interpolating statistical features of unlabeled data into the CRFs model. Wang et al. (2011) proposed a semi-supervised pipeline S&T model by incorporating  $n$ -gram and lexicon features derived from unlabeled data. Different from their concern, our emphasis is to learn the semi-supervised model by injecting the label information from a similarity graph constructed from labeled and unlabeled data.

The induction method of the proposed approach

also differs from other semi-supervised CRFs algorithms. Jiao et al. (2006), extended by Mann and McCallum (2007), reported a semi-supervised CRFs model which aims to guide the learning by minimizing the conditional entropy of unlabeled data. The proposed approach regularizes the CRFs by the graph information. Subramanya et al. (2010) proposed a graph-based self-train style semi-supervised CRFs algorithm. In the proposed approach, an analogous way of graph construction intuition is applied. But overall, our approach differs in three important aspects: first, novel feature templates are defined for measuring the similarity between vertices. Second, the critical property, i.e., sparsity, is considered among label propagation. And third, the derived label information from the graph is smoothed into the model by optimizing a modified objective function.

## 3 Background

### 3.1 Supervised Character-based Model

The character-based joint S&T approach is operated as a sequence labeling fashion that each Chinese character, i.e., *hanzi*, in the sequence is assigned with a tag. To perform segmentation and tagging simultaneously in a uniform framework, according to Ng and Low (2004), the tag is composed of a word boundary part, and a POS part, e.g., “B\_NN” refers to the first character in a word with POS tag “NN”. In this paper, 4 word boundary tags are employed: B (beginning of a word), M (middle part of a word), E (end of a word) and S (single character). As for the POS tag, we shall use the 33 tags in the Chinese tree bank. Thus, the potential composite tags of joint S&T consist of 132 ( $4 \times 33$ ) classes.

The first-order CRFs model (Lafferty et al., 2001) has been the most common one in this task. Given a set of labeled examples  $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^l$ , where  $x_i = x_i^1 x_i^2 \dots x_i^N$  is the sequence of characters in the  $i$ th sentence, and  $y_i = y_i^1 y_i^2 \dots y_i^N$  is the corresponding label sequence. The goal is to learn a CRFs model in the form,

$$p(y_i|x_i; \Lambda) = \frac{1}{Z(x_i; \Lambda)} \exp\left\{\sum_{j=1}^N \sum_{k=1}^K \lambda_k f_k(y_i^{j-1}, y_i^j, x_i, j)\right\} \quad (1)$$

where  $Z(x_i; \Lambda)$  is the partition function that normalizes the exponential form to be a probability distribution, and  $f_k(y_i^{j-1}, y_i^j, x_i, j)$ . In this study,

the baseline feature templates of joint S&T are the ones used in (Ng and Low, 2004; Jiang et al., 2008), as shown in Table 1.  $\Lambda = \{\lambda_1 \lambda_2 \dots \lambda_K\} \in \mathbb{R}^K$  are the weight parameters to be learned. In supervised training, the aim is to estimate the  $\Lambda$  that maximizes the conditional likelihood of the training data while regularizing model parameters:

$$\mathcal{L}(\Lambda) = \sum_{i=1}^l \log p(y_i|x_i; \Lambda) - R(\Lambda) \quad (2)$$

$R(\Lambda)$  can be any standard regularizer on parameters, e.g.,  $R(\Lambda) = \|\Lambda\| / 2\delta^2$ , to limit overfitting on rare features and avoid degeneracy in the case of correlated features. This objective function can be optimized by the stochastic gradient method or other numerical optimization methods.

Type	Font Size
Unigram	$C_n(n = -2, -1, 0, 1, 2)$
Bigram	$C_n C_{n+1}(n = -2, -1, 0, 1)$
Date, Digit and Alphabetic Letter	$T(C_{-2})T(C_{-1})T(C_0)$ $T(C_1)T(C_2)$

Table 1: The feature templates of joint S&T.

### 3.2 Graph-based Label Propagation

Graph-based label propagation, a critical subclass of semi-supervised learning (SSL), has been widely used and shown to outperform other SSL methods (Chapelle et al., 2006). Most of these algorithms are transductive in nature, so they cannot be used to predict an unseen test example in the future (Belkin et al., 2006). Typically, graph-based label propagation algorithms are run in two main steps: graph construction and label propagation. The graph construction provides a natural way to represent data in a variety of target domains. One constructs a graph whose vertices consist of labeled and unlabeled examples. Pairs of vertices are connected by weighted edges which encode the degree to which they are expected to have the same label (Zhu et al., 2003). Popular graph construction methods include  $k$ -nearest neighbors ( $k$ -NN) (Bentley, 1980; Beygelzimer et al., 2006),  $b$ -matching (Jebara et al., 2009) and local reconstruction (Daitch et al., 2009). Label propagation operates on the constructed graph. The primary objective is to propagate labels from a few labeled vertices to the entire graph by optimizing a loss function based on the constraints or

properties derived from the graph, e.g., smoothness (Zhu et al., 2003; Subramanya et al., 2010; Talukdar et al., 2008), or sparsity (Das and Smith, 2012). State-of-the-art label propagation algorithms include LP-ZGL (Zhu et al., 2003), Adsorption (Baluja et al., 2008), MAD (Talukdar and Crammer, 2009) and Sparse Inducing Penalties (Das and Smith, 2012).

## 4 Method

The emphasis of this work is on building a joint S&T model based on two different kinds of data sources, labeled and unlabeled data. In essence, this learning problem can be treated as incorporating certain gainful information, e.g., prior knowledge or label constraints, of unlabeled data into the supervised model. The proposed approach employs a transductive graph-based label propagation method to acquire such gainful information, i.e., label distributions from a similarity graph constructed over labeled and unlabeled data. Then, the derived label distributions are injected as virtual evidences for guiding the learning of CRFs.

### Algorithm 1 semi-supervised joint S&T induction

#### Input:

$$\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^l \text{ labeled sentences}$$

$$\mathcal{D}_u = \{(x_i)\}_{i=l+1}^{l+u} \text{ unlabeled sentences}$$

#### Output:

$\Lambda$ : a set of feature weights

- 1: **Begin**
- 2:  $\{\mathcal{G}\} = \text{construct\_graph}(\mathcal{D}_l, \mathcal{D}_u)$
- 3:  $\{q_0\} = \text{init\_labelDist}(\{\mathcal{G}\})$
- 4:  $\{q\} = \text{propagate\_label}(\{\mathcal{G}\}, \{q_0\})$
- 5:  $\{\Lambda\} = \text{train\_crf}(\mathcal{D}_l \cup \mathcal{D}_u, \{q\})$
- 6: **End**

The model induction includes the following steps (see Algorithm 1): firstly, given labeled and unlabeled data, i.e.,  $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^l$  with  $l$  labeled sentences and  $\mathcal{D}_u = \{(x_i)\}_{i=l+1}^{l+u}$  with  $u$  unlabeled sentences, a specific similarity graph  $\mathcal{G}$  representing  $\mathcal{D}_l$  and  $\mathcal{D}_u$  is constructed (**construct\_graph**). The vertices (Section 4.1) in the constructed graph consist of all *trigrams* that occur in labeled and unlabeled sentences, and edge weights between vertices are computed using the cosine distance between pointwise mutual information (PMI) statistics. Afterwards, the estimated label distributions  $q_0$  of vertices in the graph  $\mathcal{G}$  are randomly initialized (**init\_labelDist**). Subsequently,

the label propagation procedure (**propagate\_label**) is conducted for projecting label distributions  $q$  from labeled vertices to the entire graph, using the algorithm of Sparse-Inducing Penalties (Das and Smith, 2012) (Section 4.2). The final step (**train\_crf**) of the induction is incorporating the inferred trigram-level label distributions  $q$  into CRFs model (Section 4.3).

#### 4.1 Graph Construction

In most graph-based label propagation tasks, the final effect depends heavily on the quality of the graph. Graph construction thus plays a central role in graph-based label propagation (Zhu et al., 2003). For character-based joint S&T, unlike the unstructured learning problem whose vertices are formed directly by labeled and unlabeled instances, the graph construction is non-trivial. Das and Petrov (2011) mentioned that taking individual characters as the vertices would result in various ambiguities, whereas the similarity measurement is still challenging if vertices corresponding to entire sentences.

This study follows the intuitions of graph construction from Subramanya et al. (2010) in which vertices are represented by character trigrams occurring in labeled and unlabeled sentences. Formally, given a set of labeled sentences  $\mathcal{D}_l$ , and unlabeled ones  $\mathcal{D}_u$ , where  $\mathcal{D} \triangleq \{\mathcal{D}_l, \mathcal{D}_u\}$ , the goal is to form an undirected weighted graph  $\mathcal{G} = (V, E)$ , where  $V$  is defined as the set of vertices which covers all trigrams extracted from  $\mathcal{D}_l$  and  $\mathcal{D}_u$ . Here,  $V = V_l \cup V_u$ , where  $V_l$  refers to trigrams that occurs at least once in labeled sentences and  $V_u$  refers to trigrams that occur only in unlabeled sentences. The edges  $E \in V_l \times V_u$ , connect all the vertices. This study makes use of a symmetric  $k$ -NN graph ( $k = 5$ ) and the edge weights are measured by a symmetric similarity function (Equation (3)):

$$w_{i,j} = \begin{cases} \text{sim}(x_i, x_j) & \text{if } j \in \mathcal{K}(i) \text{ or } i \in \mathcal{K}(j) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where  $\mathcal{K}(i)$  is the set of the  $k$  nearest neighbors of  $x_i$  ( $|\mathcal{K}(i)| = k, \forall i$ ) and  $\text{sim}(x_i, x_j)$  is a similarity measure between two vertices. The similarity is computed based on the co-occurrence statistics over the features in Table 2. Most features we adopted are selected from those of (Subramanya et al., 2010). Note that a novel feature in the last row encodes the classes of surrounding character-

s, where four types are defined: number, punctuation, alphabetic letter and other. It is especially helpful for the graph to make connections with trigrams that may not have been seen in labeled data but have similar label information. The pointwise mutual information values between the trigrams and each feature instantiation that they have in common are summed to sparse vectors, and their cosine distances are computed as the similarities.

Description	Feature
Trigram + Context	$x_1x_2x_3x_4x_5$
Trigram	$x_2x_3x_4$
Left Context	$x_1x_2$
Right Context	$x_4x_5$
Center Word	$x_3$
Trigram - Center Word	$x_2x_4$
Left Word + Right Context	$x_2x_4x_5$
Right Word + Left Context	$x_1x_2x_3$
Type of Trigram: number, punctuation, alphabetic letter and other	$t(x_2)t(x_3)t(x_4)$

Table 2: Features employed to measure the similarity between two vertices, in a given text “ $x_1x_2x_3x_4x_5$ ”, where the trigram is “ $x_2x_3x_4$ ”.

The nature of the similarity graph enforces that the connected trigrams with high weight appearing in different texts should have similar syntax configurations. Thus, the constructed graph is expected to provide additional information that cannot be expressed directly in a sequence model (Subramanya et al., 2010). One primary benefit of this property is on enriching vocabulary coverage. In other words, the new features of various trigrams only occurring in unlabeled data can be discovered. As the excerpt in Figure 1 shows, the trigram “天津港” (Tianjin port) has no any label information, as it only occurs in unlabeled data, but fortunately its neighborhoods with similar syntax information, e.g., “上海港” (Shanghai port), “广州港” (Guangzhou port), can assist to infer the correct tag “M<sub>NN</sub>”.

#### 4.2 Label Propagation

In order to induce trigram-level label distributions from the graph constructed by the previous step, a label propagation algorithm, Sparsity-Inducing Penalties, proposed by Das and Smith (2012), is employed. This algorithm is used because it captures the property of sparsity that only a few labels

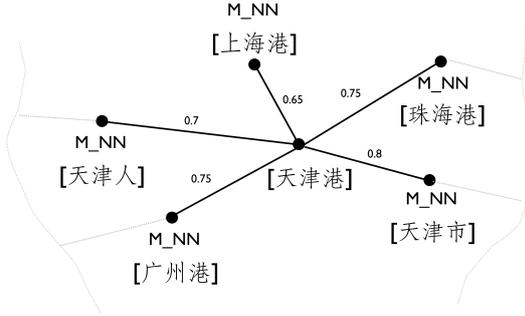


Figure 1: An excerpt from the similarity graph over trigrams on labeled and unlabeled data.

are typically associated with a given instance. In fact, the sparsity is also a common phenomenon among character-based CWS and POS tagging. The following convex objective is optimized on the similarity graph in this case:

$$\begin{aligned} & \operatorname{argmin}_q \sum_{j=1}^l \|q_j - r_j\|^2 \\ & + \mu \sum_{i=1, k \in \mathcal{N}(i)}^{l+u} w_{ik} \|q_i - q_k\|^2 + \lambda \sum_{i=1}^{l+u} \|q_i\|^2 \\ & \text{s.t. } q_i \geq 0, \forall i \in V \end{aligned} \quad (4)$$

where  $r_j$  denotes empirical label distributions of labeled vertices, and  $q_i$  denotes unnormalized estimate measures in every vertex. The  $w_{ik}$  refers to the similarity between the  $i$ th trigram and the  $k$ th trigram, and  $\mathcal{N}(i)$  is a set of neighbors of the  $i$ th trigram.  $\mu$  and  $\lambda$  are two hyperparameters whose values are discussed in Section 5. The squared-loss criterion<sup>1</sup> is used to formulate the objective function. The first term in Equation (4) is the seed match loss which penalizes the estimated label distributions  $q_j$ , if they go too far away from the empirical labeled distributions  $r_j$ . The second term is the edge smoothness loss that requires  $q_i$  should be smooth with respect to the graph, such that two vertices connected by an edge with high weight should be assigned similar labels. The final term is a regularizer to incorporate the prior knowledge, e.g., uniform distributions used in (Talukdar et al., 2008; Das and Smith, 2011). This study applies the squared norm of  $q$  to encourage sparsity per vertex. Note that the estimated label distribution

<sup>1</sup>It can be seen as a multi-class extension of quadratic cost criterion (Bengio et al., 2006) or as a variant of the objective in (Zhu et al., 2003). An entropic distance measure could also be used, e.g., KL-divergence (Subramanya et al., 2010; Das and Smith, 2012).

$q_i$  in Equation (4) is relaxed to be unnormalized, which simplifies the optimization. Thus, the objective function can be optimized by L-BFGS-B (Zhu et al., 1997), a generic quasi-Newton gradient-based optimizer. The partial derivatives of Equation (4) are computed for each parameter of  $q$  and then passed on to the optimizer that updates them such that Equation (4) is maximized.

### 4.3 Semi-Supervised CRFs Training

The trigram-level label distributions inferred in the propagation step can be viewed as a kind of valuable “prior knowledge” to regularize the learning on unlabeled data. The final step of the induction is thus to incorporate such prior knowledge into CRFs. Li (2009) generalizes the use of virtual evidence to undirected graphical models and, in particular, to CRFs for incorporating external knowledge. By extending the similar intuition, as illustrated in Figure 2, we modify the structure of a regular linear-chain CRFs on unlabeled data for smoothing the derived label distributions, where virtual evidences, i.e.,  $q$  in our case, are donated by  $\{v_1, v_2, \dots, v_T\}$ , in parallel with the state variables  $\{y_1, y_2, \dots, y_T\}$ . The modified CRFs model allows us to flexibly define the interaction between estimated state values and virtual evidences by potential functions. Therefore, given labeled and unlabeled data, the learning objective is defined as follows:

$$\mathcal{L}(\Lambda) + \sum_{i=l+1}^{l+u} E_{p(y_i|x_i, v_i; \Lambda^g)} [\log p(y_i, v_i|x_i; \Lambda)] \quad (5)$$

where the conditional probability in the second term is denoted as

$$\begin{aligned} p(y_i, v_i|x_i; \Lambda) = & \frac{1}{Z'(x_i; \Lambda)} \exp\left\{ \sum_{j=1}^N \sum_{k=1}^K \lambda_k f_k(y_i^{j-1}, y_i^j, x_i, j) \right. \\ & \left. + \alpha \sum_{t=1}^N s(y_i^t, v_i^t) \right\} \end{aligned} \quad (6)$$

The first term in Equation (5) is the same as Equation (2), which is the traditional CRFs learning objective function on the labeled data. The second term is the expected conditional likelihood of unlabeled data. It is directed to maximize the conditional likelihood of hidden states with the derived label distributions on unlabeled data, i.e.,  $p(y, v|x)$ , where  $y$  and  $v$  are jointly modeled but

the probability is still conditional on  $x$ . Here,  $Z'(x; \Lambda)$  is the partition function of normalization that is achieved by summing the numerator over both  $y$  and  $v$ . A virtual evidence feature function of  $s(y_i^t, v_i^t)$  with pre-defined weight  $\alpha$  is defined to regularize the conditional distributions of states over the derived label distributions. The learning is impacted by the derived label distributions as Equation (7): firstly, if the trigram  $x_i^{t-1}x_i^tx_i^{t+1}$  at current position does have no corresponding derived label distributions ( $v_i^t = null$ ), the value of zero is assigned to all state hypotheses so that the posteriors would not be affected by the derived information. Secondly, if it does have a derived label distribution, since the virtual evidence in this case is a distribution instead of a specific label, the label probability in the distribution under the current state hypothesis is assigned. This means that the values of state variables are constrained to agree with the derived distributions.

$$s(y_i^t, v_i^t) = \begin{cases} q_{x_i^{t-1}x_i^tx_i^{t+1}}(y_i^t) & \text{if } v_i^t \neq null \\ 0 & \text{else} \end{cases} \quad (7)$$

The second term in Equation (5) can be optimized by using the expectation maximization (EM) algorithm in the same fashion as in the generative approach, following (Li, 2009). One can iteratively optimize the  $Q$  function  $Q(\Lambda) = \sum_y p(y_i|x_i; \Lambda^g) \log p(y_i, v_i|x_i; \Lambda)$ , in which  $\Lambda^g$  is the model estimated from the previous iteration. Here the gradient of the  $Q$  function can be measured by:

$$\frac{\partial Q(\Lambda)}{\partial \Lambda_k} = \sum_t \sum_{y_i^{t-1}, y_i^t} f_k(y_i^{t-1}, y_i^t, x_i, t) \cdot (p(y_i^{t-1}, y_i^t|x_i, v_i; \Lambda) - p(y_i^{t-1}, y_i^t|x_i; \Lambda)) \quad (8)$$

The forward-backward algorithm is used to measure  $p(y_i^{t-1}, y_i^t|x_i, v_i; \Lambda)$  and  $p(y_i^{t-1}, y_i^t|x_i; \Lambda)$ . Thus, the objective function Equation (5) is optimized as follows: for the instances  $i = 1, 2, \dots, l$ , the parameters  $\Lambda$  are learned as the supervised manner; for the instances  $i = l + 1, l + 2, \dots, u + l$ , in the E-step, the expected value of  $Q$  function is computed, based on the current model  $\Lambda^g$ . In the M-step, the posteriors are fixed and updated  $\Lambda$  that maximizes Equation (5).

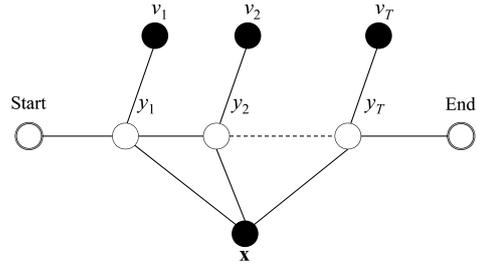


Figure 2: Modified linear-chain CRFs integrating virtual evidences on unlabeled data.

## 5 Experiment

### 5.1 Setting

The experimental data are mainly taken from the Chinese tree bank (CTB-7) and Microsoft Research (MSR)<sup>2</sup>. CTB-7 consists of over one million words of annotated and parsed text from Chinese newswire, magazine news, various broadcast news and broadcast conversation programs, web newsgroups and weblogs. It is a segmented, POS tagged<sup>3</sup> and fully bracketed corpus. The train, development and test sets<sup>4</sup> from CTB-7 and their corresponding statistics are reported in Table 3. To satisfy the characteristic of the semi-supervised learning problem, the train set, i.e., the labeled data, is formed by a relatively small amount of annotated texts sampled from CTB-7. For the unlabeled data in this experiment, a greater amount of texts is extracted from CTB-7 and MSR, which contains 53,108 sentences with 2,418,690 characters.

The performance measurement indicators for word segmentation and POS tagging (joint S&T) are balance F-score,  $F = 2PR/(P+R)$ , the harmonic mean of precision (P) and recall (R), and out-of-vocabulary recall (OOV-R). For segmentation, a token is regarded to be correct if its boundaries match the ones of a word in the gold standard. For the POS tagging, it is correct only if both the boundaries and the POS tags are perfect matches.

The experimental platform is implemented based on two toolkits: Mallet (McCallum and Kachites, 2002) and Junto (Talukdar and Pereira, 2010). Mallet is a java-based package for statistical natural language processing, which includes the CRFs implementation. Junto is a graph-

<sup>2</sup>It can be download at: [www.sighan.org/bakeoff2005](http://www.sighan.org/bakeoff2005).

<sup>3</sup>There is a total of 33 POS tags in CTB-7.

<sup>4</sup>The extracted sentences in train, development and test set were assigned with the composite tags as described in Section 3.1.

based label propagation toolkit that provides several state-of-the-art algorithms.

Data	#Sent	#Word	#Char	#OOV
Train	17,968	374,697	596,360	
Develop	1,659	46,637	79,283	0.074
Test	2,037	65,219	104,502	0.089

Table 3: Training, development and testing data.

## 5.2 Baseline and Proposed Models

In the experiment, the baseline supervised pipeline and joint S&T models are built only on the train data. The proposed model will also be compared with the semi-supervised pipeline S&T model described in (Wang et al., 2011). In addition, two state-of-the-art semi-supervised CRFs algorithms, Jiao’s CRFs (Jiao et al., 2006) and Subramanya’s CRFs (Subramanya et al., 2010), are also used to build joint S&T models. The corresponding settings of the above candidates are listed below:

- **Baseline I:** a supervised CRFs pipeline S&T model. The feature templates are from Zhao et al. (2006) and Wu et al. (2008).
- **Wang’s model:** a semi-supervised CRFs pipeline S&T model. The same feature templates in (Wang et al., 2011) are used, i.e., “+ $n$ -gram+cluster+lexicon”.
- **Baseline II:** a supervised CRFs joint S&T model. The feature templates introduced in Section 3.1 are used.
- **Jiao’s model:** a semi-supervised CRFs joint S&T model trained using the entropy regularization (ER) criteria (Jiao et al., 2006). The optimization method proposed by Mann and McCallum (2007) is applied.
- **Subramanya’s model:** a self-train style semi-supervised CRFs joint S&T model based on the same parameters used in (Subramanya et al., 2010).
- **Our model:** several parameters in our model are needed to tune based on the development set, e.g.,  $\mu$ ,  $\lambda$  and  $\alpha$ .

In all the CRFs models above, the Gaussian regularizer and stochastic gradient descent method are employed.

## 5.3 Main Results

This experiment yielded a similarity graph that consists of 462,962 trigrams from labeled and unlabeled data. The majority (317,677 trigrams) occurred only in unlabeled data. Based on the development data, the hyperparameters of our model were tuned among the following settings: for the graph propagation,  $\mu \in \{0.2, 0.5, 0.8\}$  and  $\lambda \in \{0.1, 0.3, 0.5, 0.8\}$ ; for the CRFs training,  $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ . The best performed joint settings are  $\mu = 0.5$ ,  $\lambda = 0.3$  and  $\alpha = 0.7$ . With the chosen set of hyperparameters, the test data was used to measure the final performance.

Model	Segmentation		POS Tagging	
	F <sub>1</sub>	OOV-R	F <sub>1</sub>	OOV-R
Baseline I	94.27	60.12	91.08	51.72
Wang’s	95.17	63.10	91.64	53.29
Baseline II	95.14	61.52	91.61	52.29
Jiao’s	95.58	63.05	92.11	53.27
Subramanya’s	96.30	67.12	92.46	57.15
Our model	<b>96.85</b>	<b>68.09</b>	<b>92.89</b>	<b>58.36</b>

Table 4: The performance of segmentation and POS tagging on testing data.

Table 4 summarizes the performance of segmentation and POS tagging on the test data, in comparison with the other five models. Firstly, as expected, for the two supervised baselines, the joint model outperforms the pipeline one, especially on segmentation. It obtains 0.92% and 2.32% increase in terms of F-score and OOV-R respectively. This outcome verifies the commonly accepted fact that the joint model can substantially improve the pipeline one, since POS tags provide additional information to word segmentation (Ng and Low, 2004). Secondly, it is also noticed that all four semi-supervised models are able to benefit from unlabeled data and greatly improve the results with respect to the baselines. On the whole, for segmentation, they achieve average improvements of 1.02% and 6.8% in F-score and OOV-R; whereas for POS tagging, the average increments of F-score and OOV-R are 0.87% and 6.45%. An interesting phenomenon is found among the comparisons with baselines that the supervised joint model (Baseline II) is even competitive with semi-supervised pipeline one (Wang et al., 2011). This illustrates the effects of error propagation in the pipeline approach. Thirdly, in what concerns the semi-supervised approaches, the three joint S&T models, i.e., Jiao’s, Subramanya’s and our model, are superior to the pipeline model, i.e., Wang’s

model. Moreover, the two graph-based approaches, i.e., Subramanya’s and our model, outperform the others. Most importantly, the boldface numbers in the last row illustrate that our model does achieve the best performance. Overall, for word segmentation, it obtains average improvements of 1.43% and 8.09% in F-score and OOV-R over others; for POS tagging, it achieves average improvements of 1.09% and 7.73%.

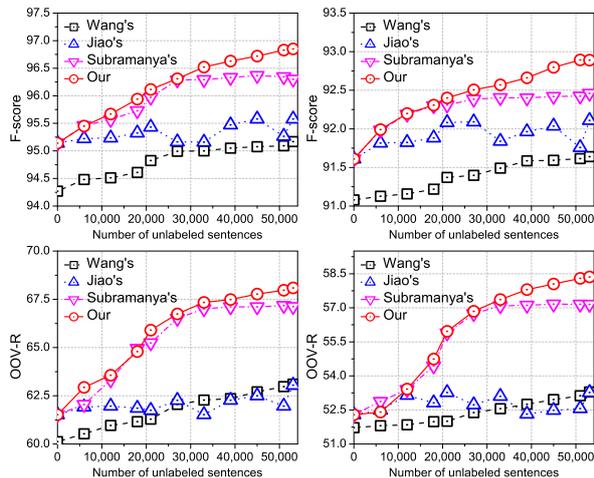


Figure 3: The learning curves of semi-supervised models on unlabeled data, where left graphs are segmentation and the right ones are tagging.

#### 5.4 Learning Curve

An additional experiment was conducted to investigate the impact of unlabeled data for the four semi-supervised models. Figure 3 illustrates the curves of F-score and OOV-R for segmentation and tagging respectively, as the unlabeled data size is progressively increased in steps of 6,000 sentences. It can be clearly observed that all curves of our model are able to mount up steadily and achieve better gains over others consistently. The most competitive performance of the other three candidates is achieved by Subramanya’s model. This strongly reveals that the knowledge derived from the similarity graph does effectively strengthen the model. But in Subramanya’s model, when the unlabeled size ascends to approximately 30,000 sentences the curves become nearly asymptotic. The semi-supervised pipeline model, Wang’s model, presents a much slower growth on all curves over the others and also begins to overfit with large unlabeled data sizes (>25,000 sentences). The figure also shows an erratic fluctuation of Jiao’s model. Since this approach aims

at minimizing conditional entropy over unlabeled data and encourages finding putative labelings for unlabeled data, it results in a data-sensitive model (Li et al., 2009).

#### 5.5 Analysis & Discussion

A statistical analysis of the segmentation and tagging results of the supervised joint model (Baseline II) and our model is carried out to comprehend the influence of the graph-based semi-supervised behavior. For word segmentation, the most significant improvement of our model is mainly concentrated on two kinds of words which are known for their difficulties in terms of CWS: a) named entities (NE), e.g., “天津港” (Tianjin port) and “保税区” (free tax zone); and b) Chinese numbers (CN), e.g., “八点五亿” (eight hundred and fifty million) and “百分之七十二” (seventy two percent). Very often, these words do not exist in the labeled data, so the supervised model is hard to learn their features. Part of these words, however, may occur in the unlabeled data. The proposed semi-supervised approach is able to discover their label information with the help of a similarity graph. Specifically, it learns the label distributions from similar words (neighborhoods), e.g., “上海港” (Shanghai port), “保护区” (protection zone), “九点七亿” (nine hundred and seventy million). The statistics in Table 5 demonstrate significant error reductions of 50.44% and 48.74% on test data, corresponding to NE and CN respectively.

Type	#word	#baErr	#gbErr	ErrDec%
NE	471	226	112	50.44
CN	181	119	61	48.74

Table 5: The statistics of segmentation error for named entities (NE) and Chinese numbers (CN) in test data. #baErr and #gbErr denote the count of segmentations by Baseline II and our model; ErrDec% denotes the error reduction.

On the other hand, to better understand the tagging results, we summarize the increase and decrease of the top five common tagging error patterns of our model over Baseline II for the correctly segmented words, as shown in Table 6. The error pattern is defined by “A→B” that refers the true tag of “A” is annotated by a tag of “B”. The obvious improvement brought by our model occurs with the tags “NN”, “CD”, “NR”, “JJ” and “NR”, where errors are reduced 60.74% on aver-

Pattern	#baErr	↓	Pattern	#baErr	↑
NN→VV	58	38	NN→NR	13	6
CD→NN	41	27	IJ→ON	9	5
NR→VV	29	17	VV→NN	4	3
JJ→NN	18	11	NR→NN	1	3
NR→VA	19	10	JJ→AD	1	2

Table 6: The statistics of POS tagging error patterns in test data. #baErr denote the count of tagging error by Baseline II, while ↓ and ↑ denotes the number of error reduced or increased by our model.

age. More impressively, there is a large portion of fixed error pattern instances stemming from OOV words. Meanwhile, it is also observed that the disambiguation of error patterns in the right portion of the table slightly suffers from our approach. In reality, it is impossible and unrealistic to request a model to be “no harms but only benefits” under whatever circumstances.

## 6 Conclusion

This study introduces a novel semi-supervised approach for joint Chinese word segmentation and POS tagging. The approach performs the semi-supervised learning in the way that the trigram-level distributions inferred from a similarity graph are used to regularize the learning of CRFs model on labeled and unlabeled data. The empirical results indicate that the similarity graph information and the incorporation manner of virtual evidences present a positive effect to the model induction.

## Acknowledgments

The authors are grateful to the Science and Technology Development Fund of Macau and the Research Committee of the University of Macau for the funding support for our research, under the reference No. 017/2009/A and RG060/09-10S/CS/FST. The authors also wish to thank the anonymous reviewers for many helpful comments.

## References

Shumeet Baluja, Rohan Seth, D. Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravich, and Mohamed Aly. 2008. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of WWW*, pages 895-904, Beijing, China.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani.

2006. Manifold regularization. *Journal of machine learning research*, 7:2399–2434.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. 2006. Label propagation and quadratic criterion. *MIT Press*.

Jon Louis Bentley. 1980. Multidimensional divide-and-conquer. *Communications of the ACM*, 23(4):214–229.

Alina Beygelzimer, Sham Kakade, and John Langford. 2006. Cover trees for nearest neighbor. In *Proceedings of ICML*, pages 97-104, New York, USA

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. 2006. Semi-supervised learning. *MIT Press*.

Samuel I. Daitch, Jonathan A. Kelner, and Daniel A. Spielman. 2009. Fitting a graph to vector data. In *Proceedings of ICML*, 201-208, NY, USA.

Dipanjan Das and Noah A. Smith. 2011. Semi-supervised framesemantic parsing for unknown predicates. In *Proceedings of ACL*, pages 1435-1444, Portland, Oregon, USA.

Dipanjan Das and Slav Petrov. 2011. Unsupervised Part-of-Speech Tagging with Bilingual Graph-based Projections. In *Proceedings of ACL*, pages 1435-1444, Portland, Oregon, USA.

Dipanjan Das and Noah A. Smith. 2012. Graph-based lexicon expansion with sparsity-inducing penalties. In *Proceedings of NAACL*, pages 677-687, Montréal, Canada.

Tony Jebara, Jun Wang, and Shih-Fu Chang. 2009. Graph construction and b-matching for semi-supervised learning. In *Proceedings of ICML*, 441-448, New York, USA.

Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Liu. 2008. A Cascaded Linear Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of ACL*, pages 897-904, Columbus, Ohio.

Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic Adaptation of Annotation Standards: Chinese Word Segmentation and POS Tagging – A Case Study. In *Proceedings of the ACL and the 4th IJCNLP of the AFNLP*, pages 522–530, Suntec, Singapore.

Feng Jiao, Shaojun Wang, and Chi-Hoon Lee. 2006. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *In Proceedings of ACL*, pages 209–216, Sydney, Australia.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun’ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of ACL and IJCNLP of the AFNLP*, pages 513- 521, Suntec, Singapore August.

- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional Random Field: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of ICML*, pages 282-289, Williams College, USA.
- Xiao Li. 2009. On the use of virtual evidence in conditional random fields. In *Proceedings of EMNLP*, pages 1289-1297, Singapore.
- Xiao Li, Ye-Yi Wang, and Alex Acero. 2009. Extracting structured information from user queries with semi-supervised conditional random fields. In *Proceedings of ACM SIGIR*, pages 572-579, Boston, USA.
- Gideon S. Mann and Andrew McCallum. 2007. Efficient computation of entropy gradient for semi-supervised conditional random fields. In *Proceedings of NAACL*, pages 109-112, New York, USA.
- McCallum and Andrew Kachites. 2002. MALLET: A Machine Learning for Language Toolkit. Software at <http://mallet.cs.umass.edu>.
- Tetsuji Nakagawa and Kiyotaka Uchimoto. 2007. A hybrid approach to word segmentation and POS tagging. In *Proceedings of ACL Demo and Poster Session*, pages 217-220, Prague, Czech Republic.
- Hwee Tou Ng and Jin Kiat Low. 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In *Proceedings of EMNLP*, Barcelona, Spain.
- Xian Qian and Yang Liu. 2012. Joint Chinese Word Segmentation, POS Tagging and Parsing. In *Proceedings of EMNLP-CoNLL*, pages 501-511, Jeju Island, Korea.
- Yanxin Shi and Mengqiu Wang. 2007. A dual-layer CRF based joint decoding method for cascade segmentation and labelling tasks. In *Proceedings of IJCAI*, Hyderabad, India.
- Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of EMNLP*, pages 167-176, Massachusetts, USA.
- Weiwei Sun. 2011. A Stacked Sub-Word Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In *Proceedings of ACL*, pages 1385-1394, Portland, Oregon.
- Weiwei Sun and Jia Xu. 2011. Enhancing Chinese word segmentation using unlabeled data. In *Proceedings of EMNLP*, pages 970-979, Scotland, UK.
- Partha Pratim Talukdar, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly Supervised Acquisition of Labeled Class Instances using Graph Random Walks. In *Proceedings of EMNLP*, pages 582-590, Hawaii, USA.
- Partha Pratim Talukdar and Koby Crammer. 2009. New Regularized Algorithms for Transductive Learning. In *Proceedings of ECML-PKDD*, pages 442 - 457, Bled, Slovenia.
- Partha Pratim Talukdar and Fernando Pereira. 2010. Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *Proceedings of ACL*, pages 1473-1481, Uppsala, Sweden.
- Yiyou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of IJCNLP*, pages 309-317, Chiang Mai, Thailand.
- Yu-Chieh Wu Jie-Chi Yang, and Yue-Shi Lee. 2008. Description of the NCU Chinese Word Segmentation and Part-of-Speech Tagging for SIGHAN Bake-off. In *Proceedings of the SIGHAN Workshop on Chinese Language Processing*, pages 161-166, Hyderabad, India.
- Jia Xu, Jianfeng Gao, Kristina Toutanova, and Hermann Ney. 2008. Bayesian semi-supervised chinese word segmentation for statistical machine translation. In *Proceedings of COLING*, pages 1017-1024, Manchester, UK.
- Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of EMNLP*, pages 888-896, Columbus, Ohio.
- Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of EMNLP*, pages 843-852, Massachusetts, USA.
- Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective tag set selection in Chinese word segmentation via conditional random field modeling. In *Proceedings of PACLIC*, pages 87-94, Wuhan, China.
- Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of ICML*, pages 912-919, Washington DC, USA.
- Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. 1997. L-BFGS-B: Fortran subroutines for large scale bound constrained optimization. *ACM Transactions on Mathematical Software*, 23:550-560.

# An Infinite Hierarchical Bayesian Model of Phrasal Translation

**Trevor Cohn**

Department of Computer Science  
The University of Sheffield  
Sheffield, United Kingdom  
t.cohn@sheffield.ac.uk

**Gholamreza Haffari**

Faculty of Information Technology  
Monash University  
Clayton, Australia  
reza@monash.edu

## Abstract

Modern phrase-based machine translation systems make extensive use of word-based translation models for inducing alignments from parallel corpora. This is problematic, as the systems are incapable of accurately modelling many translation phenomena that do not decompose into word-for-word translation. This paper presents a novel method for inducing phrase-based translation units directly from parallel data, which we frame as learning an inverse transduction grammar (ITG) using a recursive Bayesian prior. Overall this leads to a model which learns translations of entire sentences, while also learning their decomposition into smaller units (phrase-pairs) recursively, terminating at word translations. Our experiments on Arabic, Urdu and Farsi to English demonstrate improvements over competitive baseline systems.

## 1 Introduction

The phrase-based approach (Koehn et al., 2003) to machine translation (MT) has transformed MT from a narrow research topic into a truly useful technology to end users. Leading translation systems (Chiang, 2007; Koehn et al., 2007; Marcu et al., 2006) all use some kind of multi-word translation unit, which allows translations to be produced from large canned units of text from the training corpus. Larger phrases allow for the lexical context to be considered in choosing the translation, and also limit the number of reordering decisions required to produce a full translation.

Word-based translation models (Brown et al., 1993) remain central to phrase-based model training, where they are used to infer word-level alignments from sentence aligned parallel data, from

which phrasal translation units are extracted using a heuristic. Although this approach demonstrably works, it suffers from a number of shortcomings. Firstly, many phrase-based phenomena which do not decompose into word translations (e.g., idioms) will be missed, as the underlying word-based alignment model is unlikely to propose the correct alignments. Secondly, the relationship between different phrase-pairs is not considered, such as between single word translations and larger multi-word phrase-pairs or where one large phrase-pair subsumes another.

This paper develops a phrase-based translation model which aims to address the above shortcomings of the phrase-based translation pipeline. Specifically, we formulate translation using inverse transduction grammar (ITG), and seek to learn an ITG from parallel corpora. The novelty of our approach is that we develop a Bayesian prior over the grammar, such that a nonterminal becomes a ‘cache’ learning each production *and* its complete yield, which in turn is recursively composed of its child constituents. This is closely related to adaptor grammars (Johnson et al., 2007a), which also generate full tree rewrites in a monolingual setting. Our model learns translations of entire sentences while also learning their decomposition into smaller units (phrase-pairs) recursively, terminating at word translations. The model is richly parameterised, such that it can describe phrase-based phenomena while also explicitly modelling the relationships between phrase-pairs and their component expansions, thus ameliorating the disconnect between the treatment of words versus phrases in the current MT pipeline. We develop a Bayesian approach using a Pitman-Yor process prior, which is capable of modelling a diverse range of geometrically decaying distributions over infinite event spaces (here translation phrase-pairs), an approach shown to be state of the art for language modelling (Teh, 2006).

We are not the first to consider this idea; Neubig et al. (2011) developed a similar approach for learning an ITG using a form of Pitman-Yor adaptor grammar. However Neubig et al.’s work was flawed in a number of respects, most notably in terms of their heuristic beam sampling algorithm which does not meet either of the Markov Chain Monte Carlo criteria of ergodicity or detailed balance. Consequently their approach does not constitute a valid Bayesian model. In contrast, this paper provides a more rigorous and theoretically sound method. Moreover our approach results in consistent translation improvements across a number of translation tasks compared to Neubig et al.’s method, and a competitive phrase-based baseline.

## 2 Related Work

Inversion transduction grammar (or ITG) (Wu, 1997) is a well studied synchronous grammar formalism. Terminal productions of the form  $X \rightarrow e/f$  generate a word in two languages, and non-terminal productions allow phrasal movement in the translation process. Straight productions, denoted by their non-terminals inside square brackets [...], generate their symbols in the given order in both languages, while inverted productions, indicated by angled brackets  $\langle \dots \rangle$ , generate their symbols in the reverse order in the target language.

In the context of machine translation, ITG has been explored for statistical word alignment in both unsupervised (Zhang and Gildea, 2005; Cherry and Lin, 2007; Zhang et al., 2008; Pauls et al., 2010) and supervised (Haghighi et al., 2009; Cherry and Lin, 2006) settings, and for decoding (Petrov et al., 2008). Our paper fits into the recent line of work for jointly inducing the phrase table and word alignment (DeNero and Klein, 2010; Neubig et al., 2011). The work of DeNero and Klein (2010) presents a *supervised* approach to this problem, whereas our work is *unsupervised* hence more closely related to Neubig et al. (2011) which we describe in detail below.

A number of other approaches have been developed for learning phrase-based models from bilingual data, starting with Marcu and Wong (2002) who developed an extension to IBM model 1 to handle multi-word units. This pioneering approach suffered from intractable inference and moreover, suffers from degenerate solutions (DeNero and Klein, 2010). Our approach is similar to these previous works, except that we impose

additional constraints on how phrase-pairs can be tiled to produce a sentence pair, and moreover, we seek to model the embedding of phrase-pairs in one another, something not considered by this prior work. Another strand of related research is in estimating a broader class of synchronous grammars than ITGs, such as SCFGs (Blunsom et al., 2009b; Levenberg et al., 2012). Conceptually, our work could be readily adapted to general SCFGs using similar techniques.

This work was inspired by adaptor grammars (Johnson et al., 2007a), a monolingual grammar formalism whereby a non-terminal rewrites in a single step as a complete subtree. The model prior allows for trees to be generated as a mixture of a cache and a base adaptor grammar. In our case, we have generalised to a bilingual setting using an ITG. Additionally, we have extended the model to allow recursive nesting of adapted non-terminals, such that we end up with an infinitely recursive formulation where the top-level and base distributions are explicitly linked together.

As mentioned above, ours is not the first work attempting to generalise adaptor grammars for machine translation; (Neubig et al., 2011) also developed a similar approach based around ITG using a Pitman-Yor Process prior. Our approach improves upon theirs in terms of the model and inference, and critically, this is borne out in our experiments where we show uniform improvements in translation quality over a baseline system, as compared to their almost entirely negative results. We believe that their approach had a number of flaws: For inference they use a beam-search, which may speed up processing but means that they are no longer sampling from the true distribution, nor a distribution with the same support as the posterior. Moreover they include a Metropolis-Hastings correction step, which is required to correct the samples to account for repeated substructures which will be otherwise underrepresented. Consequently their approach does not constitute a Markov Chain Monte Carlo sampler, but rather a complex heuristic.

The other respect in which this work differs from Neubig et al. (2011) is in terms of model formulation. They develop an ITG which generates phrase-pairs as terminals, while we employ a more restrictive word-based model which forces the decomposition of every phrase-pair. This is an important restriction as it means that we jointly learn

a word and phrase based model, such that word based phenomena can affect the phrasal structures. Finally our approach models separately the three different types of ITG production (monotone, swap and lexical emission), allowing for a richer parameterisation which the model exploits by learning different hyper-parameter values.

### 3 Model

The generative process of the model follows that of ITG with the following simple grammar

$$\begin{aligned} X &\rightarrow [X X] \mid \langle X X \rangle \\ X &\rightarrow e/f \mid e/\perp \mid \perp/f, \end{aligned}$$

where  $[\cdot]$  denotes monotone ordering and  $\langle \cdot \rangle$  denotes a swap in one language. The symbol  $\perp$  denotes the empty string. This corresponds to a simple generative story, with each stage being a non-terminal rewrite starting with  $X$  and terminating when there are no frontier non-terminals.

A popular variant is a *phrasal ITG*, where the leaves of the ITG tree are phrase-pairs and the training seeks to learn a segmentation of the source and target which yields good phrases. We would not expect this model to do very well as it cannot consider overlapping phrases, but instead is forced into selecting between many competing – and often equally viable – options. Our approach improves over the phrasal model by recursively generating complete phrases. This way we don't insist on a single tiling of phrases for a sentence pair, but explicitly model the set of hierarchically nested phrases as defined by an ITG derivation. This approach is closer in spirit to the phrase-extraction heuristic, which defines a set of 'atomic' terminal phrase-pairs and then extracts every combination of these atomic phase-pairs which is contiguous in the source and target.<sup>1</sup>

The generative process is that we draw a complete ITG tree,  $t \sim P_2(\cdot)$ , as follows:

1. choose the rule type,  $r \sim R$ , where  $r \in \{\text{mono}, \text{swap}, \text{emit}\}$
2. for  $r = \text{mono}$ 
  - (a) draw the complete subtree expansion,  $t = X \rightarrow [\dots] \sim T_M$
3. for  $r = \text{swap}$ 
  - (a) draw the complete subtree expansion,  $t = X \rightarrow \langle \dots \rangle \sim T_S$

<sup>1</sup>Our technique considers the subset of phrase-pairs which are consistent with the ITG tree.

4. for  $r = \text{emit}$ 
  - (a) draw a pair of strings,  $(e, f) \sim E$
  - (b) set  $t = X \rightarrow e/f$

Note that we split the problem of drawing a tree into two steps: first choosing the top-level rule type and then drawing a rule of that type. This gives us greater control than simply drawing a tree of any type from one distribution, due to our parameterisation of the priors over the model parameters  $T_M$ ,  $T_S$  and  $E$ .

To complete the generative story, we need to specify the prior distributions for  $T_M$ ,  $T_S$  and  $E$ . First, we deal with the emission distribution,  $E$  which we draw from a Dirichlet Process prior  $E \sim \text{DP}(b_E, P_0)$ . We restrict the emission rules to generate word pairs rather than phrase pairs.<sup>2</sup> For the base distribution,  $P_0$ , we use a simple uniform distribution over word pairs,

$$P_0(e, f) = \begin{cases} \eta^2 \frac{1}{\sqrt{E} \sqrt{F}} & e \neq \perp, f \neq \perp \\ \eta(1-\eta) \frac{1}{\sqrt{F}} & e = \perp, f \neq \perp \\ \eta(1-\eta) \frac{1}{\sqrt{E}} & e \neq \perp, f = \perp \end{cases},$$

where the constant  $\eta$  denotes the binomial probability of a word being aligned.<sup>3</sup>

We use Pitman-Yor Process priors for the  $T_M$  and  $T_S$  parameters

$$\begin{aligned} T_M &\sim \text{PYP}(a_M, b_M, P_1(\cdot | r = \text{mono})) \\ T_S &\sim \text{PYP}(a_S, b_S, P_1(\cdot | r = \text{swap})) \end{aligned}$$

where  $P_1(t_1, t_2 | r)$  is a distribution over a pair of trees (the left and right children of a monotone or swap production).  $P_1$  is defined as follows:

1. choose the complete left subtree  $t_1 \sim P_2$ ,
2. choose the complete right subtree  $t_2 \sim P_2$ ,
3. set  $t = X \rightarrow [t_1 t_2]$  or  $t = X \rightarrow \langle t_1 t_2 \rangle$  depending on  $r$

This generative process is mutually recursive:  $P_2$  makes draws from  $P_1$  and  $P_1$  makes draws from  $P_2$ . The recursion is terminated when the rule type  $r = \text{emit}$  is drawn.

Following standard practice in Bayesian models, we integrate out  $R$ ,  $T_M$ ,  $T_S$  and  $E$ . This means draws from  $P_2$  (or  $P_1$ ) are no longer *iid*: for any non-trivial tree, computing its probability under this model is complicated by the fact

<sup>2</sup>Note that we could allow phrases here, but given the model can already reason over phrases by way of its hierarchical formulation, this is an unnecessary complication.

<sup>3</sup>We also experimented with using word translation probabilities from IBM model 1, based on the prior used by Levenberg et al. (2012), however we found little empirical difference compared with this simpler uniform model.

that the probability of its two subtrees are interdependent. This is best understood in terms of the Chinese Restaurant Franchise (CRF; Teh et al. (2006)), which describes the posterior distribution after integrating out the model parameters. In our case we can consider the process of drawing a tree from  $P_2$  as a customer entering a restaurant and choosing where to sit, from an infinite set of tables. The seating decision is based on the number of other customers at each table, such that popular tables are more likely to be joined than unpopular or empty ones. If the customer chooses an occupied table, the identity of the tree is then set to be the same as for the other customers also seated there. For empty tables the tree must be sampled from the base distribution  $P_1$ . In the standard CRF analogy, this leads to another customer entering the restaurant one step up in the hierarchy, and this process can be chained many times. In our case, however, every new table leads to new customers reentering the original restaurant – these correspond to the left and right child trees of a monotone or swap rule. The recursion terminates when a table is shared, or a new table is labelled with a emit rule.

### 3.1 Inference

The probability of a tree (i.e., a draw from  $P_2$ ) under the model is

$$P_2(t) = P(r)P_2(t|r) \quad (1)$$

where  $r$  is the rule type, one of `mono`, `swap` or `emit`. The distribution over types,  $P(r)$ , is defined as

$$P(r) = \frac{n_r^{T,-} + b_T \frac{1}{3}}{n^{T,-} + b_T}$$

where  $n^{T,-}$  are the counts over rules of types.<sup>4</sup>

The second component in (1),  $P_2(t|r)$ , is defined separately for each rule type. For  $r = \text{mono}$  or  $r = \text{swap}$  rules, it is defined as

$$P_2(t|r) = \frac{n_{t,r}^- - K_{t,r}^- a_r}{n_r^- + b_r} + \frac{K_r^- a_r + b_r}{n_r^- + b_r} P_1(t_1, t_2|r), \quad (2)$$

where  $n_{t,r}^-$  is the count for tree  $t$  in the other training sentences,  $K_{t,r}^-$  is the table count for  $t$  and  $n_r^-$

<sup>4</sup>The conditioning on event and table counts,  $n^-$ ,  $K^-$  is omitted for clarity.

and  $K_r^-$  are the total count of trees and tables, respectively. Finally, the probability for  $r = \text{emit}$  is given by

$$P_2(t|r = \text{emit}) = \frac{n_{t,E}^- + b_E P_0(e, f)}{n_r^- + b_r},$$

where  $t = X \rightarrow e/f$ .

To complete the derivation we still need to define  $P_1$ , which is formulated as

$$P_1(t_1, t_2) = P_2(t_1)P_2(t_2|t_1),$$

where the conditioning of the second recursive call to  $P_2$  reflects that the counts  $n^-$  and  $K^-$  may be affected by the first draw from  $P_2$ . Although these two draws are assumed *iid* in the prior, after marginalising out  $T$  they are no longer independent. For this reason, evaluating  $P_2(t)$  is computationally expensive, requiring tracking of repeated substructures in descendent sub-trees of  $t$ , which may affect other descendants. This results in an asymptotic complexity exponential in the number of nodes in the tree. For this reason we consider trees annotated with binary values denoting their table assignment, namely whether they share a table or are seated alone. Given this, the calculation is greatly simplified, and has linear complexity.<sup>5</sup>

We construct an approximating ITG following the technique used for sampling trees from monolingual tree-substitution grammars (Cohn et al., 2010). To do so we encode the first term from (2) separately from the second term (corresponding to draws from  $P_1$ ). Summing together these two alternate paths – i.e., during inside inference – we recover  $P_2$  as shown in (2). The full grammar transform for inside inference is shown in Table 1.

The sampling algorithm closely follows the process for sampling derivations from Bayesian PCFGs (Johnson et al., 2007b). For each sentence-pair, we first decrement the counts associated with its current tree, and then sample a new derivation. This involves first constructing the inside lattice using the productions in Table 1, and then performing a top-down sampling pass. After sampling each derivation from the approximating grammar, we then convert this into its corresponding ITG tree, which we then score with the full model and accept or reject the sample using the

<sup>5</sup>To support this computation, we track explicit table assignments for every training tree and their component subtrees. We also sample trees labelled with seating indicator variables.

Type	$X \rightarrow M$	$P(r = \text{mono})$
	$X \rightarrow S$	$P(r = \text{swap})$
	$X \rightarrow E$	$P(r = \text{emit})$
Base	$M \rightarrow [XX]$	$\frac{K_M^- a_M + b_M}{n_M^- + b_M}$
	$S \rightarrow \langle XX \rangle$	$\frac{K_S^- a_S + b_S}{n_S^- + b_S}$
Count	For every tree, $t$ , of type $r = \text{mono}$ , with $n_{t,M} > 0$ :	
	$M \rightarrow \text{sig}(t)$	$\frac{n_{t,M}^- - K_{t,M}^- a_r}{n_M^- + b_M}$
	$\text{sig}(t) \rightarrow \text{yield}(t)$	1
	For every tree, $t$ , of type $r = \text{swap}$ , with $n_{t,S} > 0$ :	
$S \rightarrow \text{sig}(t)$	$\frac{n_{t,S}^- - K_{t,S}^- a_S}{n_S^- + b_S}$	
$\text{sig}(t) \rightarrow \text{yield}(t)$	1	
Emit	For every word pair, $e/f$ in sentence pair, where one of $e, f$ can be $\perp$ :	
	$E \rightarrow e/f$	$P_2(t)$

Table 1: Grammar transformation rules for MAP inside inference. The function  $\text{sig}(t)$  returns a unique identifier for the complete tree  $t$ , and the function  $\text{yield}(t)$  returns the pair of terminal strings from the yield of  $t$ .

Metropolis-Hastings algorithm.<sup>6</sup> Accepted samples then replace the old tree (otherwise the old tree is retained) and the model counts are incremented. This process is then repeated for each sentence pair in the corpus in a random order.

## 4 Experiments

**Datasets** We train our model across three language pairs: Urdu→English (UR-EN), Farsi→English (FA-EN), and Arabic→English (AR-EN). The corpora statistics of these translation tasks are summarised in Table 2. The UR-EN corpus comes from NIST 2009 translation evaluation.<sup>7</sup> The AR-EN training data consists of the eTIRR corpus (LDC2004E72), the Arabic news corpus (LDC2004T17), the Ummah corpus (LDC2004T18), and the sentences with confidence  $c > 0.995$  in the ISI automatically extracted web parallel corpus (LDC2006T02). For FA-EN, we use TEP<sup>8</sup> Tehran English-Persian Parallel corpus (Pilevar and Faili, 2011), which consists of conversational/informal text extracted

<sup>6</sup>The full model differs from the approximating grammar in that it accounts for inter-dependencies between subtrees by recursively tracking the changes in the customer and table counts while scoring the tree. Around 98% of samples were accepted in our experiments.

<sup>7</sup><http://www.itl.nist.gov/iad/mig/tests/mt/2009>

<sup>8</sup><http://ece.ut.ac.ir/NLP/resources.htm>

	source	target	sentences
UR-EN	745K	575K	148K
FA-EN	4.7M	4.4M	498K
AR-EN	1.94M	2.08M	113K

Table 2: Corpora statistics showing numbers of parallel sentences and source and target words for the training sets.

from 1600 movie subtitles. We tokenized this corpus, removed noisy single-word sentences, randomly selected the development and test sets, and used the rest of the corpus as the training set. We discard sentences with length above 30 from the datasets for all experiments.<sup>9</sup>

**Sampler configuration** Samplers are initialised with trees created from GIZA++ alignments constructed using a SCFG factorisation method (Blunsom et al., 2009a). This algorithm represents the translation of a sentence as a large SCFG rule, which it then factorises into lower rank SCFG rules, a process akin to rule binarisation commonly used in SCFG decoding. Rules that cannot be reduced to a rank-2 SCFG are simplified by dropping alignment edges until they can be factorised, the net result being an ITG derivation largely respecting the alignments.<sup>10</sup>

The blocked sampler was run 1000 iterations for UR-EN, 100 iterations for FA-EN and AR-EN. After each full sampling iteration, we resample all the hyper-parameters using slice-sampling, with the following priors:  $a \sim \text{Beta}(1, 1)$ ,  $b \sim \text{Gamma}(10, 0.1)$ . Figure 1 shows the posterior probability improves with each full sampling iterations. The alignment probability was set to  $\eta = 0.99$ . The sampling was repeated for 5 independent runs, and we present results where we combine the outputs of these runs. This is a form of Monte Carlo integration which allows us to represent the uncertainty in the posterior, while also representing multiple modes, if present.

The time complexity of our inference algorithm is  $O(n^6)$ , which can be prohibitive for large scale machine translation tasks. We reduce the complexity by constraining the *inside* inference to consider only derivations which are compatible

<sup>9</sup>Hence the BLEU scores we get for the baselines may appear lower than what reported in the literature.

<sup>10</sup>Using the factorised alignments directly in a translation system resulted in a slight loss in BLEU versus using the un-factorised alignments. Our baseline system uses the latter.

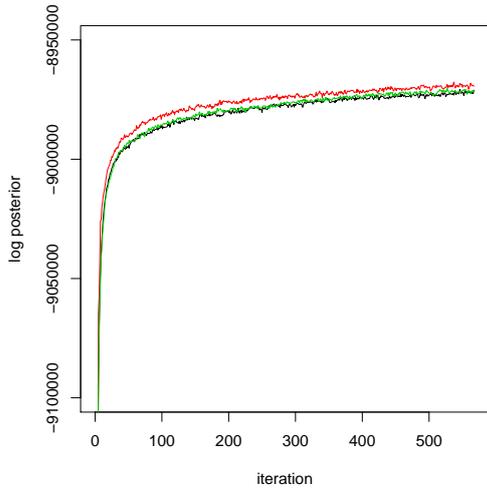


Figure 1: Training progress on the UR-EN corpus, showing the posterior probability improving with each full sampling iteration. Different colours denote independent sampling runs.

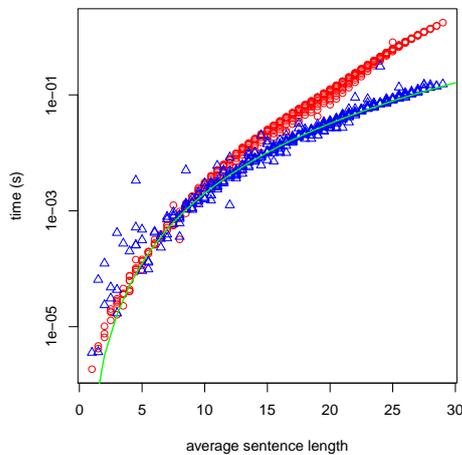


Figure 2: The runtime cost of bottom-up inside inference and top-down sampling as a function of sentence length (UR-EN), with time shown on a logarithmic scale. Full ITG inference is shown with red circles, and restricted inference using the intersection constraints with blue triangles. The average time complexity for the latter is roughly  $O(l^4)$ , as plotted in green  $t = 2 \times 10^{-7}l^4$ .

with high confidence alignments from GIZA++.<sup>11</sup> Figure 2 shows the sampling time with respect to the average sentence length, showing that our alignment-constrained sampling algorithm is better than the unconstrained algorithm with empirical complexity of  $n^4$ . However, the time complexity is still high, so we set the maximum sentence length to 30 to keep our experiments practicable. Presumably other means of inference may be more efficient, such as Gibbs sampling (Levenberg et al., 2012) or auxiliary variable sampling (Blunsom and Cohn, 2010); we leave these extensions to future work.

**Baselines.** Following (Levenberg et al., 2012; Neubig et al., 2011), we evaluate our model by using its output word alignments to construct a phrase table. As a baseline, we train a phrase-based model using the Moses toolkit<sup>12</sup> based on the word alignments obtained using GIZA++ in both directions and symmetrized using the growdiag-final-and heuristic<sup>13</sup> (Koehn et al., 2003). This alignment is used as input to the rule factorisation algorithm, producing the ITG trees with which we initialise our sampler. To put our results in the context of the previous work, we also compare against *pialign* (Neubig et al., 2011), an ITG algorithm using a Pitman-Yor process prior, as described in Section 2.<sup>14</sup>

In the end-to-end MT pipeline we use a standard set of features: relative-frequency and lexical translation model probabilities in both directions; distance-based distortion model; language model and word count. We set the distortion limit to 6 and max-phrase-length to 7 in all experiments. We train 3-gram language models using modified Kneser-Ney smoothing. For AR-EN experiments the language model is trained on English data as (Blunsom et al., 2009a), and for FA-EN and UR-EN the English data are the target sides of the bilingual training data. We use minimum error rate training (Och, 2003) with nbest list size 100 to optimize the feature weights for maximum development BLEU.

<sup>11</sup>These are taken from the final model 4 word alignments, using the intersection of the source-target and target-source models. These alignments are very high precision (but have low recall), and therefore are unlikely to harm the model.

<sup>12</sup><http://www.statmt.org/moses>

<sup>13</sup>We use the default parameter settings in both Moses and GIZA++.

<sup>14</sup><http://www.phontron.com/pialign>

		Baselines		This paper	
		GIZA++	pialign	individual	combination
UR-EN		16.95	15.65	16.68 ± .12	<b>16.97</b>
FA-EN		20.69	21.41	21.36 ± .17	<b>21.50</b>
AR-EN	MT03	44.05	43.30	44.8 ± .28	<b>45.10</b>
	MT04	38.15	37.78	38.4 ± .08	<b>38.4</b>
	MT05	42.81	42.18	43.13 ± .23	<b>43.45</b>
	MT08	32.43	<b>33.00</b>	32.7 ± .15	32.80

Table 3: The BLEU scores for the translation tasks of three language pairs. The individual column show the average and 95% confidence intervals for 5 independent runs, whereas the combination column show the results for combining the phrase tables of all these runs. The baselines are GIZA++ alignments and those generated by the pialign (Neubig et al., 2011) **bold**: the best result.

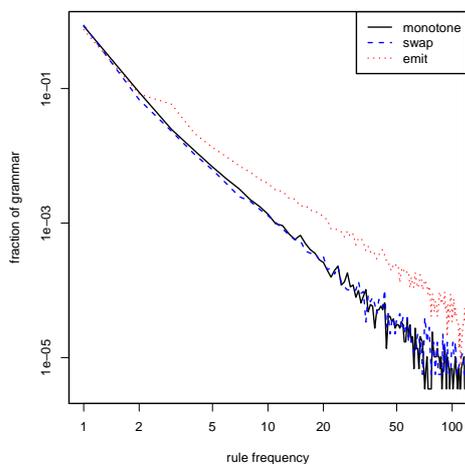


Figure 3: Fraction of rules with a given frequency, using a single sample grammar (UR-EN).

#### 4.1 Results

Table 3 shows the BLEU scores for the three translation tasks UR/AR/FA→EN based on our method against the baselines. For our models, we report the average BLEU score of the 5 independent runs as well as that of the *aggregate* phrase table generated by these 5 independent runs. There are a number of interesting observations in Table 3. Firstly, combining the phrase tables from independent runs results in increased BLEU scores, possibly due to the representation of uncertainty in the outputs, and the representation of different modes captured by the individual models. We believe this type of Monte Carlo model averaging should be considered in general when sampling techniques are employed for grammatical inference, e.g. in parsing and translation. Secondly, our approach consistently improves over the Giza++ baseline often by a large margin, whereas *pialign* under-

performs the GIZA++ baseline in many cases. Thirdly, our model consistently outperforms *pialign* (except in AR-EN MT08 which is very close). This highlights the modeling and inference differences between our method and the *pialign*.

## 5 Analysis

In this section, we present some insights about the learned grammar and the model hyper-parameters. Firstly, we start by presenting various statistics about different learned grammars. Figure 3 shows the fraction of rules with a given frequency for each of the three rule types. The three types of rule exhibit differing amounts of high versus low frequency rules, and all roughly follow power laws. As expected, there is a higher tendency to reuse high-frequency emissions (or single-word translation) compared to other rule types, which are the basic building blocks to compose larger rules (or phrases). Table 4 lists the high frequency monotone and swap rules in the learned grammar. We observe the high frequency swap rules capture reordering in verb clusters, preposition-noun inversions and adjective-noun reordering. Similar patterns are seen in the monotone rules, along with some common canned phrases. Note that “in Iraq” appears twice, once as an inversion in UR-EN and another time in monotone order for AR-EN.

Secondly, we analyse the values learned for the model hyper-parameters; Figure 4.(a) shows the posterior distribution over the hyper-parameter values. There is very little spread in the inferred values, suggesting the sampling chains may have converged. Furthermore, there is a large difference between the learned hyper-parameters for the monotone rules versus the swap rules. For the Pitman-Yor Process prior, the values of the hyper-

FARSI-ENGLISH	
pialign	sure/mTm n,chief/r ys, you sure/mTm ny, im sure/mTm nm, boss/r ys, make sure/mTm n, are you sure/mTm ny, anyway/hr HAL, president/r ys jmhwr, not sure/mTm n nystm, im sure/mTm nm kh, i'm sure/mTm nm, sure/mTm nA
our method	sure/mTm } , have/dA \$ th, be/b \$, have/dA \$ th bA \$, let me/* Ar, because of/xATr, sure/mTm } n, do/kAr rA, come on/zwd bA, excuse me/bbx, kill/rA bk, come on/zwdbA, more than/\$ tr, behind/p \$, what do/mnZwrt, what do you/mnZwrt , kill/k \$, dont worry/ngrAn nbA, is it/\$ dh, welcome/xw \$  , chief/r } ys, make sure/mTm, is/my \$, make sure/mTm } , make sure/mTm} n, im sorry/bbx, left/g * A, if/Agr \$
ARABIC-ENGLISH	
pialign	said /ال. قال. -, states/المتحدة, united/الولايات, al-wafd/الوفد, efforts/بذل, of mass destruction/الدمار الشامل, youm/اليوم, jintao/جين تاو, alam al youm/العلم اليوم, al-ittihad, /الاتحاد, the field of/الجال, islamabad/اسلام, scheduled/المقرر, al-alam al-youm/العلم اليوم, وزراء, prime/الوقت نفسه, meanwhile/شبه الجزيرة, peninsula/شبه الجزيرة, al-hayat / ص, / al-hayat, / دعوة من, / al-hayat, / الدعوة من, korean peninsula/الجزيرة الكورية, al-nahar "النداء", department/وزارة الخارجية, cote/كوت, as possible/ممكن, وقت, al alam al youm/العلم اليوم, al-alam al-youm, /العلم اليوم, at the invitation/دعوة من, jacques/جاك, well as/كذلك, points/نقاط, vladimir putin/فلاديمير بوتين, george w. bush/جورج
our method	the united/المتحدة, us dollars/امريكي, prime/الرئيس, china 'الصين', spokesman/المتحدث, many/من, is expected/المتوقع, is expected to/المتوقع, at least/على الأقل, on tuesday/يوم, egypt 'مصر', thursday/يوم, the un/المتحدة, on thursday/يوم, friday/يوم, on friday/يوم, to/حسب, al-wafd /الوفد, the us/المتحدة, for/لـ, الانسبة لـ, first time/الاولى, further/من, iraq 'العراق', israeli prime/الرئيس الاسرائيلي, the two/البلدين, on saturday/يوم, on sunday/يوم, u.s./المتحدة, views/النظر, sharon 'شارون', country 'البلد', he said/ذلك, israel 'اسرائيل', people 'الشعب', here/هنا, china 'الصين', he said/اضاف, earlier/وقت, china 'الصين', at least/على الأقل, the u.s./المتحدة, the gaza/قطاع, the gaza strip/قطاع, are expected/المتوقع, are expected to/المتوقع, are expected to/المتوقع, million u.s./مليون, according/وفقا, to/تواصل, order/من, in order/من, he pointed/اشار, mfa , asharq al /الاورس, mfa , asharq al awsat/الاورس, arafat 'عزفات'

Table 5: Good phrase pairs in the top-100 high frequency phrase pairs specific to the phrase tables coming from our method vs that of pialign for FA-EN and AR-EN translation tasks.

parameters affects the rate at which the number of types grows compared to the number of tokens. Specifically, as the discount  $a$  or the concentration  $b$  parameters increases we expect for a relative increase in the number of types. If the number of observed monotone and swap rules were equal, then there would be a higher chance in reusing the monotone rules. However, the number of observed monotone and swap rules are not equal, as plotted in Figure 4.(b). Similar results were observed for the other language pairs (figures omitted for space reasons).

Thirdly, we performed a manual evaluation for the quality of the phrase-pairs learned exclusively by our method vs *pialign*. For each method, we considered the top-100 high frequency phrase-pairs which are specific to that method. Then we asked a bilingual human expert to identify reasonably well phrase-pairs among these top-100 phrase-pairs. The results are summarized in Table 5, and show that we learn roughly twice as many reasonably good phrase-pairs for AR-EN and FA-EN compared to *pialign*.

## Conclusions

We have presented a novel method for learning a phrase-based model of translation directly from parallel data which we have framed as learning an inverse transduction grammar (ITG) using a recursive Bayesian prior. This has led to a model which learns translations of entire sentences, while also learning their decomposition into smaller units (phrase-pairs) recursively, terminating at word translations. We have presented a Metropolis-Hastings sampling algorithm for blocked inference in our non-parametric ITG. Our experiments on Urdu-English, Arabic-English, and Farsi-English translation tasks all demonstrate improvements over competitive baseline systems.

## Acknowledgements

The first author was supported by the EPSRC (grant EP/I034750/1) and an Erasmus-Mundus scholarship funding a research visit to Melbourne. The second author was supported by an early career research award from Monash University.

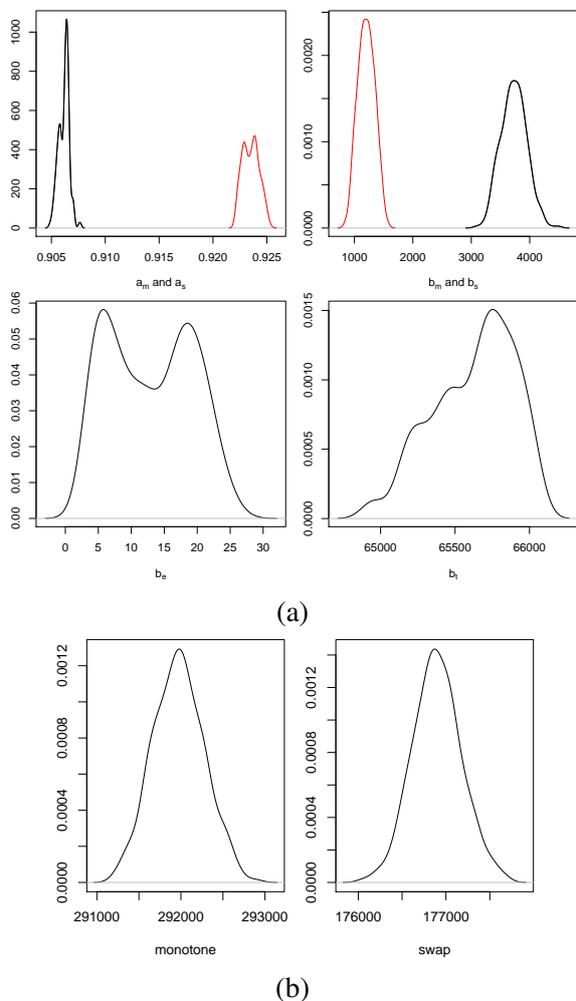


Figure 4: (a) Posterior over the hyper-parameters,  $a_M, a_S, b_M, b_S, b_E, b_T$ , measured for UR-EN using samples 400–500 for 3 independent sampling chains, and the intersection constraints. (b) Posterior over the number of monotone and swap rules in the resultant grammars. The distribution for emission rules was also peaked about 147k rules.

260	[ [ he/AnhwN ] [ $\perp$ /nY ] ] [ said/khA ] ]
222	[ < [ said/khA ] [ $\perp$ /nY ] > [ that/kh ] ]
219	[ [ [ he/AnhwN ] [ $\perp$ /nY ] ] [ said/khA ] ] [ that/kh ] ]
148	[ [ [ if/Agr ] [ $\perp$ /  ] ] [ you/p ] ]
108	[ [ he/AnhwN ] < [ said/khA ] [ $\perp$ /nY ] > ]
<i>Urdu-English</i>	
890	[ [ one/yky ] [ of/Az ] ]
843	[ < [ yeah/rh ] [ $\perp$ /  ] > [ ./ . ] ]
738	[ [ with/bA ] [ me/mn ] ]
644	[ [ [ okay/bA ] [ $\perp$ /\$ ] ] [ $\perp$ /h ] ] [ ./ . ] ]
608	[ [ to/bh ] [ me/mn ] ]
251	[ < [ is/dh ] [ it/\$ ] > ]
220	[ < [ tell/bgw ] [ me/mn ] > ]
199	[ < [ i/ $\perp$ ] [ can/twnm ] > [ 't/nmy ] >
190	[ [ [ who/ky ] [ are/hsty ] ] [ you/tw ] ]
187	[ < [ told/gft ] [ me/mn ] > ]
<i>Farsi-English</i>	
566	[ [ in/في ] [ iraq/العراق ] ]
414	[ [ in/في ] [ egypt/مصر ] ]
391	[ [ this/هذا ] [ year/عام ] ]
356	[ [ asharq/الشرق ] [ al-awsat/الاورس ] ]
300	[ [ in/في ] [ iraq/العراق ] ]
4024	[ [ ./ . ] [ " / " ] ]
1312	[ < [ the/ ] [ united/المتحدة ] > [ states/الولايات ] >
665	[ [ united/المتحدة ] [ states/الولايات ] ]
650	[ [ last/الاضري ] [ year/عام ] ]
467	[ < [ the/ ] [ united/المتحدة ] > [ nations/الامم ] >
<i>Arabic-English</i>	

Table 4: Top 5 monotone and swap productions and their counts. Rules with mostly punctuation or encoding 1:many or many:1 alignments were omitted.

## References

- Phil Blunsom and Trevor Cohn. 2010. Inducing synchronous grammars with slice sampling. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 238–241, Los Angeles, California, June. Association for Computational Linguistics.
- Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009a. A Gibbs sampler for phrasal synchronous grammar induction. In *ACL2009*, Singapore, August.
- Phil Blunsom, Trevor Cohn, and Miles Osborne. 2009b. Bayesian synchronous grammar induction. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 161–168. MIT Press.
- P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.
- Colin Cherry and Dekang Lin. 2006. Soft syntactic constraints for word alignment through discriminative training. In *Proceedings of COLING/ACL*. Association for Computational Linguistics.
- Colin Cherry and Dekang Lin. 2007. Inversion transduction grammar for joint phrasal translation modeling. In *Proc. of the HLT-NAACL Workshop on Syntax and Structure in Statistical Translation (SSST 2007)*, Rochester, USA.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Trevor Cohn, Phil Blunsom, and Sharon Goldwater. 2010. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, pages 3053–3096.
- John DeNero and Dan Klein. 2010. Discriminative modeling of extraction sets for machine translation. In *The 48th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL)*.
- Aria Haghighi, John Blitzer, and Dan Klein. 2009. Better word alignments with supervised itg models. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, Suntec, Singapore. Association for Computational Linguistics.
- Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007a. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, Cambridge, MA.
- Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007b. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proc. of the 7th International Conference on Human Language Technology Research and 8th Annual Meeting of the NAACL (HLT-NAACL 2007)*, pages 139–146.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of the 3rd International Conference on Human Language Technology Research and 4th Annual Meeting of the NAACL (HLT-NAACL 2003)*, pages 81–88, Edmonton, Canada, May.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of the 45th Annual Meeting of the ACL (ACL-2007)*, Prague.
- Abby Levenberg, Chris Dyer, and Phil Blunsom. 2012. A Bayesian model for learning SCFGs with discontinuous rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 223–232, Jeju Island, Korea, July. Association for Computational Linguistics.
- Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proc. of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pages 133–139, Philadelphia, July. Association for Computational Linguistics.
- Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proc. of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP-2006)*, pages 44–52, Sydney, Australia, July.
- Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 632–641, Portland, Oregon, USA, 6.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41st Annual Meeting of the ACL (ACL-2003)*, pages 160–167, Sapporo, Japan.
- Adam Pauls, Dan Klein, David Chiang, and Kevin Knight. 2010. Unsupervised syntactic alignment with inversion transduction grammars. In *Proceedings of the North American Conference of the Association for Computational Linguistics (NAACL)*. Association for Computational Linguistics.

- Slav Petrov, Aria Haghighi, and Dan Klein. 2008. Coarse-to-fine syntactic machine translation using language projections. In *Proceedings of EMNLP*. Association for Computational Linguistics.
- M. T. Pilevar and H. Faili. 2011. Tep: Tehran english-persian parallel corpus. In *Proc. International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*.
- Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.
- Y. W. Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.
- Hao Zhang and Daniel Gildea. 2005. Stochastic lexicalized inversion transduction grammar for alignment. In *Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.
- Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008. Bayesian learning of non-compositional phrases with synchronous parsing. In *Proc. of the 46th Annual Conference of the Association for Computational Linguistics: Human Language Technologies (ACL-08:HLLT)*, pages 97–105, Columbus, Ohio, June.

# Additive Neural Networks for Statistical Machine Translation

Lemao Liu<sup>1</sup>, Taro Watanabe<sup>2</sup>, Eiichiro Sumita<sup>2</sup>, Tiejun Zhao<sup>1</sup>

<sup>1</sup>School of Computer Science and Technology

Harbin Institute of Technology (HIT), Harbin, China

<sup>2</sup>National Institute of Information and Communication Technology (NICT)

3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan

{lmliu | tjzhao}@mtlab.hit.edu.cn

{taro.watanabe | eiichiro.sumita}@nict.go.jp

## Abstract

Most statistical machine translation (SMT) systems are modeled using a log-linear framework. Although the log-linear model achieves success in SMT, it still suffers from some limitations: (1) the features are required to be linear with respect to the model itself; (2) features cannot be further interpreted to reach their potential. A neural network is a reasonable method to address these pitfalls. However, modeling SMT with a neural network is not trivial, especially when taking the decoding efficiency into consideration. In this paper, we propose a variant of a neural network, i.e. additive neural networks, for SMT to go beyond the log-linear translation model. In addition, word embedding is employed as the input to the neural network, which encodes each word as a feature vector. Our model outperforms the log-linear translation models with/without embedding features on Chinese-to-English and Japanese-to-English translation tasks.

## 1 Introduction

Recently, great progress has been achieved in SMT, especially since Och and Ney (2002) proposed the log-linear model: almost all the state-of-the-art SMT systems are based on the log-linear model. Its most important advantage is that arbitrary features can be added to the model. Thus, it casts complex translation between a pair of languages as feature engineering, which facilitates research and development for SMT.

Regardless of how successful the log-linear model is in SMT, it still has some shortcomings.

---

This joint work was done while the first author visited NICT.

On the one hand, features are required to be linear with respect to the objective of the translation model (Nguyen et al., 2007), but it is not guaranteed that the potential features be linear with the model. This induces modeling inadequacy (Duh and Kirchhoff, 2008), in which the translation performance may not improve, or may even decrease, after one integrates additional features into the model. On the other hand, it cannot deeply interpret its surface features, and thus can not efficiently develop the potential of these features. What may happen is that a feature  $p$  does initially not improve the translation performance, but after a nonlinear operation, e.g.  $\log(p)$ , it does. The reason is not because this feature is useless but the model does not efficiently interpret and represent it. Situations such as this confuse explanations for feature designing, since it is unclear whether such a feature contributes to a translation or not.

A neural network (Bishop, 1995) is a reasonable method to overcome the above shortcomings. However, it should take constraints, e.g. the decoding efficiency, into account in SMT. Decoding in SMT is considered as the expansion of translation states and it is handled by a heuristic search (Koehn, 2004a). In the search procedure, frequent computation of the model score is needed for the search heuristic function, which will be challenged by the decoding efficiency for the neural network based translation model. Further, decoding with non-local (or state-dependent) features, such as a language model, is also a problem. Actually, even for the (log-) linear model, efficient decoding with the language model is not trivial (Chiang, 2007).

In this paper, we propose a variant of neural networks, i.e. additive neural networks (see Section 3 for details), for SMT. It consists of two components: a linear component which captures non-local (or state dependent) features and a non-linear component (i.e., neural network) which encodes lo-

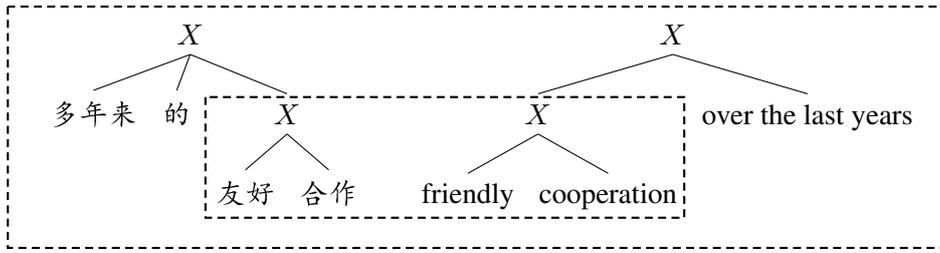


Figure 1: A bilingual tree with two synchronous rules,  $r_1 : X \rightarrow \langle \text{友好 合作}; \text{friendly cooperation} \rangle$  and  $r_2 : X \rightarrow \langle \text{多年来的 } X; X \text{ over the last years} \rangle$ . The inside rectangle denotes the partial derivation  $d_1 = \{r_1\}$  with the partial translation  $e_1 = \text{“friendly cooperation”}$ , and the outside rectangle denotes the derivation  $d_2 = \{r_1, r_2\}$  with the translation  $e_2 = \text{“friendly cooperation over the last years”}$ .

cal (or state independent) features. Compared with the log-linear model, it has more powerful expressive abilities and can deeply interpret and represent features with hidden units in neural networks. Moreover, our method is simple to implement and its decoding efficiency is comparable to that of the log-linear model. We also integrate word embedding into the model by representing each word as a feature vector (Collobert and Weston, 2008). Because of the thousands of parameters and the non-convex objective in our model, efficient training is not simple. We propose an efficient training methodology: we apply the mini-batch conjugate sub-gradient algorithm (Le et al., 2011) to accelerate the training; we also propose pre-training and post-training methods to avoid poor local minima. The biggest contribution of this paper is that it goes beyond the log-linear model and proposes a non-linear translation model instead of re-ranking model (Duh and Kirchhoff, 2008; Sokolov et al., 2012).

On both Chinese-to-English and Japanese-to-English translation tasks, experiment results show that our model can leverage the shortcomings suffered by the log-linear model, and thus achieves significant improvements over the log-linear based translation.

## 2 Log-linear Model, Revisited

### 2.1 Log-linear Translation Model

Och and Ney (2002) proposed the log-linear translation model, which can be formalized as follows:

$$\mathbf{P}(e, d|f; W) = \frac{\exp \{W^\top \cdot h(f, e, d)\}}{\sum_{e', d'} \exp \{W^\top \cdot h(f, e', d')\}}, \quad (1)$$

where  $f$  denotes the source sentence, and  $e(e')$  denotes its translation candidate;  $d(d')$  is a derivation over the pair  $\langle f, e \rangle$ , i.e.,

a collection of synchronous rules for Hiero grammar (Chiang, 2005), or phrase pairs in Moses (Koehn et al., 2007);  $h(f, e, d) = (h_1(f, e, d), h_2(f, e, d), \dots, h_K(f, e, d))^\top$  is a  $K$ -dimensional feature vector defined on the tuple  $\langle f, e, d \rangle$ ;  $W = (w_1, w_2, \dots, w_K)^\top$  is a  $K$ -dimensional weight vector of  $h$ , i.e., the parameters of the model, and it can be tuned by the toolkit MERT (Och, 2003). Different from Brown’s generative model (Brown et al., 1993), the log-linear model does not assume strong independency holds, and allows arbitrary features to be integrated into the model easily. In other words, it can transform complex language translation into feature engineering: it can achieve high translation performance if reasonable features are chosen and appropriate parameters are assigned for the weight vector.

### 2.2 Decoding By Search

Given a source sentence  $f$  and a weight  $W$ , decoding finds the best translation candidate  $\hat{e}$  via the programming problem:

$$\begin{aligned} \langle \hat{e}, \hat{d} \rangle &= \arg \max_{e, d} \mathbf{P}(e, d|f; W) \\ &= \arg \max_{e, d} \{W^\top \cdot h(f, e, d)\}. \end{aligned} \quad (2)$$

Since the range of  $\langle e, d \rangle$  is exponential with respect to the size of  $f$ , the exact decoding is intractable and an inexact strategy such as beam search is used instead in practice.

The idea of search for decoding can be shown in **Figure 1**: it encodes each search state as a partial translation together with its derivation, e.g.  $\langle e_1, d_1 \rangle$ ; it consequently expands the states from the initial (*empty*) state to the end state  $\langle e_2, d_2 \rangle$  according to the translation rules  $r_1$  and  $r_2$ . During the state expansion process, the score  $w_i \cdot$

$h_i(f, e, d)$  for a partial translation is calculated repeatedly. In the log-linear model, if  $h_i(f, e, d)$  is a local feature, the calculation of its score  $w_i \cdot h_i(f, e, d)$  has a substructure, and thus it can be calculated with dynamic programming which accelerates its decoding. For the non-local features such as the language model, Chiang (2007) proposed a cube-pruning method for efficient decoding. The main reason why cube-pruning works is that the translation model is linear and the model score for the language model is approximately monotonic (Chiang, 2007).

### 3 Additive Neural Networks

#### 3.1 Motivation

Although the log-linear model has achieved great progress for SMT, it still suffers from some pitfalls: it requires features be linear with the model and it can not interpret and represent features deeply. The neural network model is a reasonable method to overcome these pitfalls. However, the neural network based machine translation is far from easy.

As mentioned in Section 2, the decoding procedure performs an expansion of translation states. Firstly, let us consider a simple case in neural network based translation where all the features in the translation model are independent of the translation state, i.e. all the components of the vector  $h(f, e, d)$  are local features. In this way, we can easily define the following translation model with a single-layer neural network:

$$\mathbf{S}(f, e, d; W, M, B) = W^\top \cdot \sigma(M \cdot h(f, e, d) + B), \quad (3)$$

where  $M \in \mathbb{R}^{u \times K}$  is a matrix, and  $B \in \mathbb{R}^u$  is a vector, i.e. bias;  $\sigma$  is a single-layer neural network with  $u$  hidden units, i.e. an element wise sigmoid function  $\text{sigmoid}(x) = 1/(1 + \exp(-x))$ . For consistent description in the rest, we also represent Eq. (3) as a function of a feature vector  $h$ , i.e.  $\mathbf{S}(h; W, M, B) = W^\top \cdot \sigma(M \cdot h + B)$ .

Now let us consider the search procedure with the model in Eq. (3) using **Figure 1** as our example. Suppose the current translation state is encoded as  $\langle e_1, d_1 \rangle$ , which is expanded into  $\langle e_2, d_2 \rangle$  using the rule  $r_2$  ( $d_2 = d_1 \cup \{r_2\}$ ). Since  $h$  is state-independent,  $h(f, e_2, d_2) = h(f, e_1, d_1) + h(r_2)$ . However, since  $S(f, e, d; W, M, B)$  is non-decomposable as a linear model, there is no substructure for calculating  $S(f, e_2, d_2; W, M, B)$ ,

and one has to re-calculate it via Eq. (3) even if the score of  $S(f, e_1, d_1; W, M, B)$  for its previous state  $\langle e_1, d_1 \rangle$  is available. When the size of the parameter  $(W, M, B)$  is relatively large, it will be a challenge for the decoding efficiency.

In order to keep the substructure property,  $S(f, e_2, d_2; W, M, B)$  should be represented as  $F(S(f, e_1, d_1; W, M, B); S(h(r_2); M, B))$  by a function  $F$ . For simplicity, we suppose that the additive property holds in  $F$ , and then we can obtain a new translation model via the following recursive equation:

$$S(f, e_2, d_2; W, M, B) = S(f, e_1, d_1; W, M, B) + S(h(r_2); W, M, B). \quad (4)$$

Since the above model is defined only on local features, it ignores the contributions from non-local features. Actually, existing works empirically show that some non-local features, especially language model, contribute greatly to machine translation.

Scoring for non-local features such as a  $n$ -gram language model is not easily done. In log-linear translation model, Chiang (2007) proposed a cube-pruning method for scoring the language model. The premise of cube-pruning is that the language model score is approximately monotonic (Chiang, 2007). However, if scoring the language model with a neural network, this premise is difficult to hold. Therefore, one of the solutions is to preserve a linear model for scoring the language model directly.

#### 3.2 Definition

According to the above analysis, we propose a variant of a neural network model for machine translation, and we call it **Additive Neural Networks** or **AdNN** for short.

The AdNN model is a combination of a linear model and a neural network: non-local features, e.g. LM, are linearly modeled for the cube-pruning strategy, and local features are modeled by the neural network for deep interpretation and representation. Formally, the AdNN based translation model is discriminative but non-probabilistic, and it can be defined as follows:

$$S(f, e, d; \theta) = W^\top \cdot h(f, e, d) + \sum_{r \in d} W'^\top \cdot \sigma(M \cdot h'(r) + B), \quad (5)$$

where  $h$  and  $h'$  are feature vectors with dimension  $K$  and  $K'$  respectively, and each component of  $h'$  is a local feature which can be defined on a rule  $r : X \rightarrow \langle \alpha, \gamma \rangle$ ;  $\theta = (W, W', M, B)$  is the model parameters with  $M \in \mathbb{R}^{u \times K'}$ . In this paper, we focus on a single-layer neural network for its simplicity, and one can similarly define  $\sigma$  as a multi-layer neural network.

Again for the example shown in **Figure 1**, the model score defined in Eq. (5) for the pair  $\langle e_2, d_2 \rangle$  can be represented as follows:

$$S(f, e_2, d_2; \theta) = W^\top \cdot h(f, e_2, d_2) + W'^\top \cdot \sigma(M \cdot h'(r_1) + B) + W'^\top \cdot \sigma(M \cdot h'(r_2) + B).$$

Eq. (5) is similar to both additive models (Buja et al., 1989) and generalized additive neural networks (Potts, 1999): it consists of many additive terms, and each term is either a linear or a non-linear (a neural network) model. That is the reason why our model is called “additive neural networks”. Of course, our model still has some differences from both of them. Firstly, our model is decomposable with respect to rules instead of the component variables. Secondly, some of its additive terms share the same parameters  $(M, B)$ .

There are also strong relationships between AdNN and the log-linear model. If we consider the parameters  $(M, B)$  as constant and  $\sigma(M \cdot h'(r) + B)$  as a new feature vector, then AdNN is reduced to a log-linear model. Since both  $(M, B)$  and  $(W, W')$  are parameters in AdNN, our model can jointly learn the feature  $\sigma(M \cdot h'(r) + B)$  and tune the weight  $(W, W')$  of the log-linear model together. That is different from most works under the log-linear translation framework, which firstly learn features or sub-models and then tune the log-linear model including the learned features in two separate steps. By joint training, AdNN can learn the features towards the translation evaluation metric, which is the main advantage of our model over the log-linear model.

In this paper, we apply our AdNN model to hierarchical phrase based translation, and it can be similarly applied to phrase-based or syntax-based translation. Similar to Hiero (Chiang, 2005), the feature vector  $h$  in Eq. (5) includes 8 default features, which consist of translation probabilities, lexical translation probabilities, word penalty, glue rule penalty, synchronous rule penalty and language model. These default features are included

because they empirically perform well in the log-linear model. For the local feature vector  $h'$  in Eq (5), we employ word embedding features as described in the following subsection.

### 3.3 Word Embedding features for AdNN

Word embedding can relax the sparsity introduced by the lexicalization in NLP, and it improves the systems for many tasks such as language model, named entity recognition, and parsing (Collobert and Weston, 2008; Turian et al., 2010; Collobert, 2011). Here, we propose embedding features for rules in SMT by combining word embeddings.

Firstly, we will define the embedding for the source side  $\alpha$  of a rule  $r : X \rightarrow \langle \alpha, \gamma \rangle$ . Let  $\mathcal{V}_S$  be the vocabulary in the source language with size  $|\mathcal{V}_S|$ ;  $\mathbb{R}^{n \times |\mathcal{V}_S|}$  be the word embedding matrix, each column of which is the word embedding ( $n$ -dimensional vector) for the corresponding word in  $\mathcal{V}_S$ ; and  $maxSource$  be the maximal length of  $\alpha$  for all rules. We further assume that the  $\alpha$  for all rules share the same length as  $maxSource$ ; otherwise, we add  $maxSource - |\alpha|$  words “NULL” to the end of  $\alpha$  to obtain a new  $\alpha$ . We define the embedding of  $\alpha$  as the concatenation of the word embedding of each word in  $\alpha$ . In particular, for the non-terminal in  $\alpha$ , we define its word embedding as the vector whose components are 0.1; and we define the word embedding of “NULL” as  $\mathbf{0}$ . Then, we similarly define the embedding for the target side of a rule, given an embedding matrix for the target vocabulary. Finally, we define the embedding of a rule as the concatenation of the embedding of its source and target sides.

In this paper, we apply the word embedding matrices from the RNNLM toolkit (Mikolov et al., 2010) with the default settings: we train two RNN language models on the source and target sides of training corpus, respectively, and then we obtain two matrices as their by-products<sup>1</sup>. It would be potentially better to train the word embedding matrix from a much larger corpus as (Collobert and Weston, 2008), and we will leave this as a future task.

### 3.4 Decoding

Substituting the  $\mathbf{P}(e, d|f; W)$  in Eq. (2) with  $S(f, e, d; \theta)$  in Eq. (5), we can obtain its corre-

<sup>1</sup>In the RNNLM toolkit, the default dimension for word embedding is  $n = 30$ . In our experiments, the maximal length of  $\alpha$  and  $\gamma$  are 5 and 12 respectively. Thus the dimension for  $h'$  is  $K' = 30 \times (5 + 12) = 510$ .

sponding decoding formula:

$$\langle \hat{e}, \hat{d} \rangle = \arg \max_{e,d} S(f, e, d; \theta).$$

Given the model parameter  $\theta = (W, W', M, B)$ , if we consider  $(M, B)$  as constant and  $\sigma(M \cdot h'(r) + B)$  as an additional feature vector besides  $h$ , then Eq. (5) goes back to being a log-linear model with parameter  $(W, W')$ . In this way, the decoding for AdNN can share the same search strategy and cube pruning method as the log-linear model.

## 4 Training Method

### 4.1 Training Objective

For the log-linear model, there are various tuning methods, e.g. MERT (Och, 2003), MIRA (Watanabe et al., 2007; Chiang et al., 2008), PRO (Hopkins and May, 2011) and so on, which iteratively optimize a weight such that, after re-ranking a k-best list of a given development set with this weight, the loss of the resulting 1-best list is minimal. In the extreme, if the k-best list consists only of a pair of translations  $\langle \langle e^*, d^* \rangle, \langle e', d' \rangle \rangle$ , the desirable weight should satisfy the assertion: if the BLEU score of  $e^*$  is greater than that of  $e'$ , then the model score of  $\langle e^*, d^* \rangle$  with this weight will be also greater than that of  $\langle e', d' \rangle$ . In this paper, a pair  $\langle e^*, e' \rangle$  for a source sentence  $f$  is called as a preference pair for  $f$ . Following PRO, we define the following objective function under the max-margin framework to optimize the AdNN model:

$$\frac{1}{2} \|\theta\|^2 + \frac{\lambda}{N} \sum_f \sum_{e^*, d^*, e', d'} \delta(f, e^*, d^*, e', d'; \theta), \quad (6)$$

with

$$\delta(\cdot) = \max \{S(f, e', d'; \theta) - S(f, e^*, d^*; \theta) + 1, 0\}$$

where  $f$  is a source sentence in a given development set, and  $\langle \langle e^*, d^* \rangle, \langle e', d' \rangle \rangle$  is a preference pair for  $f$ ;  $N$  is the number of all preference pairs;  $\lambda > 0$  is a regularizer.

### 4.2 Optimization Algorithm

Since there are thousands of parameters in Eq. (6) and the tuning in SMT will minimize Eq. (6) repeatedly, efficient and scalable optimization methods are required. Following Le et al. (2011), we apply the mini-batch Conjugate Sub-Gradient (mini-batch CSG) method to minimize Eq. (6).

Compared with the sub-gradient descent, mini-batch CSG has some advantages: (1) it can accelerate the calculation of the sub-gradient since it calculates the sub-gradient on a subset of preference pairs (i.e. mini-batch) instead of all of the preference pairs; (2) it reduces the number of iterations since it employs the conjugate information besides the sub-gradient. Algorithm 1 shows the procedure to minimize Eq. (6).

---

#### Algorithm 1 Mini-batch conjugate subgradient

---

**Input:**  $\theta^1, T, CGIter, batch-size, k-best-list$

- 1: **for all**  $t$  such that  $1 \leq t \leq T$  **do**
- 2: Sample mini-batch preference pairs with size  $batch-size$  from  $k-best-list$
- 3: Calculate some quantities for CG, e.g. training objective  $Obj$ , subgradient  $\Delta$ , according to Eq. (6) defined over the sampled preference pairs
- 4:  $\theta^{t+1} = CG(\theta^t, Obj, \Delta, CGIter)$
- 5: **end for**

**Output:**  $\theta^{T+1}$

---

In detail, line 2 in Algorithm 1 firstly follows PRO to sample a set of preference pairs from  $k-best-list$ , and then uniformly samples  $batch-size$  pairs from the preference pair set. Line 3 calculates some quantities for CG, and Line 4 calls a CG optimizer<sup>2</sup> and obtains  $\theta^{t+1}$ . At the end of the algorithm, it returns the result  $\theta^{T+1}$ . In this work, we set the maximum number of CG iterations,  $CGIter$ , to a small number, which means  $\theta^{t+1}$  will be returned within  $CGIter$  iterations before the CG converges, for faster learning.

### 4.3 Pre-Training and Post-Training

Since Eq. (6) is non-linear, there are many local minimal solutions. Actually, this problem is inherent and is one many works based on the neural network for other NLP tasks such as language model and parsing, also suffer from. And these works empirically show that some pre-training methods, which provide a reasonable initial solution, can improve the performance. Observing the structure of Eq. (5) and the relationships between our model and a log-linear model, we propose the following simple pre-training method.

---

<sup>2</sup>In implementation, we call the CG toolkit (Hager and Zhang, 2006), which requires overloading objective and sub-gradient functions. For easier description, we substitute overloading functions and transform the value of functions in the pseudo-code.

If we set  $W' = 0$ , the model defined in Eq. (5) can be regarded as a log-linear model with features  $h$ . Therefore, we pre-train  $W$  using MERT or PRO by holding  $W' = 0$ , and use  $(W, W' = 0, M, B)$  as an initializer<sup>3</sup> for Algorithm 1.

Although the above pre-training would provide a reasonable solution, Algorithm 1 may still fall into local minima. We also propose a post-training method: after obtaining a solution with Algorithm 1, we modify this solution slightly to get a new solution. The idea of the post-training method is similar to that of the pre-training method. Suppose  $\theta = (W, W', M, B)$  be the solution obtained from Algorithm 1. If we consider both M and B to be constant, the Eq. (5) goes back to the log-linear model whose features are  $(h, \sigma(M \cdot h' + B))$  and parameters are  $(W, W')$ . Again, we train the parameters  $(W, W')$  with MERT or PRO and get the new parameters  $(\bar{W}, \bar{W}')$ . Therefore, we can set  $\theta = (\bar{W}, \bar{W}', M, B)$  as the final solution for Eq. (6). The advantage of post-training is that it optimizes a convex programming derived from the original nonlinear (non-convex) programming in Eq. (6), and thus it may decrease the risk of poor local optima.

#### 4.4 Training Algorithm

---

##### Algorithm 2 Training Algorithm

---

**Input:**  $MaxIter$ , a dev set, parameters (e.g.  $\lambda$ ) for Algorithm 1

- 1: Pre-train to obtain  $\theta_1 = (W, W' = 0, M, B)$  as the initial parameter
- 2: **for all**  $i$  such that  $1 \leq i \leq MaxIter$  **do**
- 3: Decode with  $\theta_i$  on the dev set and merge all k-best-lists
- 4: Run Algorithm 1 based on the merged k-best-list to obtain  $\theta_{i+1}$
- 5: **end for**
- 6: Post-train based on  $\theta_{MaxIter+1}$  to obtain  $\theta$

**Output:**  $\theta$

---

The whole training for the AdNN model is summarized in Algorithm 2. Given a development set, we first run pre-training to obtain an initial parameter  $\theta_1$  for Algorithm 1 in line 1. Secondly, it iteratively performs decoding and optimization for  $MaxIter$  times in the loop from line 2 to line 5: it decodes with the parameter  $\theta_i$  and merges all the

<sup>3</sup>To avoid the symmetry in the solution, we sample a very small  $(M, B)$  from the gaussian distribution in practice instead of setting  $(M, B) = 0$ .

k-best-lists in line 3; and it then runs Algorithm 1 to optimize  $\theta_{i+1}$ . Thirdly, it runs the post-training to get the result  $\theta$  based on  $\theta_{MaxIter+1}$ .

Of course, we can run post-training after running Algorithm 1 at each iteration  $i$ . However, since each pass of post-training (e.g. PRO) takes several hours because of multiple decoding times, we run it only once, at the end of the iterations instead.

## 5 Experiments and Results

### 5.1 Experimental Setting

We conduct our experiments on the Chinese-to-English and Japanese-to-English translation tasks. For the Chinese-to-English task, the training data is the FBIS corpus (news domain) with about 240k sentence pairs; the development set is the NIST02 evaluation data; the development test set is NIST05; and the test datasets are NIST06, and NIST08. For the Japanese-to-English task, the training data with 300k sentence pairs is from the NTCIR-patent task (Fujii et al., 2010); the development set, development test set, and two test sets are averagely extracted from a given development set with 4000 sentences, and these four datasets are called test1, test2, test3 and test4, respectively.

We run GIZA++ (Och and Ney, 2000) on the training corpus in both directions (Koehn et al., 2003) to obtain the word alignment for each sentence pair. Using the SRILM Toolkits (Stolcke, 2002) with modified Kneser-Ney smoothing, we train a 4-gram language model for the Chinese-to-English task on the Xinhua portion of the English Gigaword corpus and a 4-gram language model for the Japanese-to-English task on the target side of its training data. In our experiments, the translation performances are measured by case-sensitive BLEU4 metric<sup>4</sup> (Papineni et al., 2002). The significance testing is performed by paired bootstrap re-sampling (Koehn, 2004b).

We use an in-house developed hierarchical phrase-based translation (Chiang, 2005) for our baseline system, which shares the similar setting as Hiero (Chiang, 2005), e.g. beam-size=100, k-best-size=100, and is denoted as **L-Hiero** to emphasize its log-linear model. We tune L-Hiero with two methods MERT and PRO implemented in the Moses toolkit. On the same experiment settings, the performance of L-Hiero is comparable

<sup>4</sup>We use mteval-v13a.pl as the evaluation tool(Ref. <http://www.itl.nist.gov/iad/mig/tests/mt/2008/scoring.html>).

	Seconds/Sent
L-Hiero	1.77
AdNN-Hiero-E	1.88

Table 1: The decoding time comparison on NIST05 between L-Hiero and AdNN-Hiero-E.

to that of Moses: on the NIST05 test set, L-Hiero achieves 25.1 BLEU scores and Moses achieves 24.8. Further, we integrate the embedding features (See Section 3.3) into the log-linear model along with the default features as L-Hiero, which is called **L-Hiero-E**. Since L-Hiero-E has hundreds of features, we use PRO as its tuning toolkit.

**AdNN-Hiero-E** is our implementation of the AdNN model with embedding features, as discussed in Section 3, and it shares the same codebase and settings as L-Hiero. We adopt the following setting for training AdNN-Hiero-E:  $u=10$ ;  $batch-size=1000$  and  $CGiter=3$ , as referred in (Le et al., 2011), and  $T=200$  in Algorithm 1; the pre-training and post-training methods as PRO; the regularizer  $\lambda$  in Eq. (6) as 10 and 30, and  $MaxIter$  as 16 and 20 in Algorithm 2, for Chinese-to-English and Japanese-to-English tasks, respectively. Although there are several parameters in AdNN which may limit its practicality, according to many of our internal studies, most parameters are insensitive to AdNN except  $\lambda$  and  $MaxIter$ , which are common in other tuning toolkits such as MIRA and can be tuned<sup>5</sup> on a development test dataset.

Since both MERT and PRO tuning toolkits involve randomness in their implementations, all BLEU scores reported in the experiments are the average of five tuning runs, as suggested by Clark et al. (2011) for fairer comparisons. For AdNN, we report the averaged scores of five post-training runs, but both pre-training and training are performed only once.

## 5.2 Results and Analysis

As discussed in Section 3, our AdNN-Hiero-E shares the same decoding strategy and pruning method as L-Hiero. When compared with L-Hiero, decoding for AdNN-Hiero-E only needs additional computational times for the features in the hidden units, i.e.  $\sigma(M \cdot h'(r) + B)$ . Since

<sup>5</sup>For easier tuning, we tuned these two parameters on a given development test set without post-training in Algorithm 2.

Chinese-to-English				
		NIST05	NIST06	NIST08
L-Hiero	MERT	25.10 <sup>+</sup>	24.46 <sup>+</sup>	17.42 <sup>+</sup>
	PRO	25.57 <sup>+</sup>	25.27 <sup>+</sup>	18.33 <sup>+</sup>
L-Hiero-E	PRO	24.80 <sup>+</sup>	24.46 <sup>+</sup>	18.20 <sup>+</sup>
AdNN-Hiero-E		26.37	25.93	19.42
Japanese-to-English				
		test2	test3	test4
L-Hiero	MERT	24.35 <sup>+</sup>	25.62 <sup>+</sup>	23.68 <sup>+</sup>
	PRO	24.38 <sup>+</sup>	25.55 <sup>+</sup>	23.66 <sup>+</sup>
L-Hiero-E	PRO	24.47 <sup>+</sup>	25.86 <sup>+</sup>	24.03 <sup>+</sup>
AdNN-Hiero-E		25.14	26.32	24.45

Table 2: The BLEU comparisons between AdNN-Hiero-E and Log-linear translation models on the Chinese-to-English and Japanese-to-English tasks. + means the comparison is significant over AdNN-Hiero-E with  $p < 0.05$ .

these features are not dependent on the translation states, they are computed and saved to memory when loading the translation model. During decoding, we just look up these scores instead of re-calculating them on the fly. Therefore, the decoding efficiency of AdNN-Hiero-E is almost the same as that of L-Hiero. As shown in Table 1 the average decoding time for L-Hiero is 1.77 seconds/sentence while that for AdNN-Hiero-E is 1.88 seconds/sentence on the NIST05 test set.

Word embedding features can improve the performance on other NLP tasks (Turian et al., 2010), but its effect on log-linear based SMT is not as expected. As shown in Table 2, L-Hiero-E gains little over L-Hiero for the Japanese-to-English task, and even decreases the performance over L-Hiero for the Chinese-to-English task. These results further prove our claim in Section 1, i.e. the log-linear model requires the features to be linear with the model and thus limits its expressive abilities. However, after the single-layer non-linear operator (sigmoid functions) on the embedding features for deep interpretation and representation, AdNN-Hiero-E gains improvements over both L-Hiero and L-Hiero-E, as depicted in Table 2. In detail, for the Chinese-to-English task, AdNN-Hiero-E improves more than 0.6 BLEU scores over L-Hiero on both test sets: the gains over L-Hiero tuned with PRO are 0.66 and 1.09 on NIST06 and NIST08, respectively, and the gains over L-Hiero tuned with MERT are even more. Similar results are achieved on the Japanese-to-English task. AdNN-Hiero-E gains about 0.7 BLEU scores on

Chinese-to-English			
	NIST05	NIST06	NIST08
L-Hiero	25.57 <sup>+</sup>	25.27 <sup>+</sup>	18.33 <sup>+</sup>
AdNN-Hiero-E	26.37	25.93	19.42
AdNN-Hiero-D	26.21	26.07	19.54
Japanese-to-English			
	test2	test3	test4
L-Hiero	24.38	25.55	23.66
AdNN-Hiero-E	25.14 <sup>+</sup>	26.32 <sup>+</sup>	24.45 <sup>+</sup>
AdNN-Hiero-D	24.42	25.46	23.73

Table 3: The effect of different feature setting on AdNN model. + means the comparison is significant over AdNN-Hiero-D with  $p < 0.05$ .

both test sets.

In addition, to investigate the effect of different feature settings on AdNN, we alternatively design another setting for  $h'$  in Eq. (5): we use the default features for both  $h'$  and  $h$ . In particular, the language model of a rule for  $h'$  is locally calculated without the contexts out of the rule as described in (Chiang, 2007). We call the AdNN model with this setting **AdNN-Hiero-D**<sup>6</sup>. Although there are serious overlaps between  $h$  and  $h'$  for AdNN-Hiero-D which may limit its generalization abilities, as shown in Table 3, it is still comparable to L-Hiero on the Japanese-to-English task, and significantly outperforms L-Hiero on the Chinese-to-English translation task. To investigate the reason why the gains for AdNN-Hiero-D on the two different translation tasks differ, we calculate the perplexities between the target side of training data and test datasets on both translation tasks. We find that the perplexity of the 4-gram language model for the Chinese-to-English task is 321.73, but that for the Japanese-to-English task is only 81.48. Based on these similarity statistics, we conjecture that the log-linear model does not fit well for difficult translation tasks (e.g. translation task on the news domain). The problem seems to be resolved by simply alternating feature representations through non-linear models, i.e. AdNN-Hiero-D, even with single-layer networks.

## 6 Related Work

Neural networks have achieved widespread attentions in many NLP tasks, e.g. the language

<sup>6</sup>All its parameters are shared with AdNN-Hiero-E except  $\lambda$  and  $MaxIter$ , which are tuned on the development test datasets.

model (Bengio et al., 2003); POS, Chunking, NER, and SRL (Collobert and Weston, 2008); Parsing (Collobert and Weston, 2008; Socher et al., 2011); and Machine transliteration (Deselaers et al., 2009). Our work is, of course, highly motivated by these works. Unlike these works, we propose a variant neural network, i.e. additive neural networks, starting from SMT itself and taking both of the model definition and its inference (decoding) together into account.

Our variant of neural network, AdNN, is highly related to both additive models (Buja et al., 1989) and generalized additive neural networks (Potts, 1999; Waal and Toit, 2007), in which an additive term is either a linear model or a neural network. Unlike additive models and generalized additive neural networks, our model is decomposable with respect to translation rules rather than its component variables considering the decoding efficiency of machine translation; and it allows its additive terms of neural networks to share the same parameters for a compact structure to avoid sparsity.

The idea of the neural network in machine translation has already been pioneered in previous works. Castaño et al. (1997) introduced a neural network for example-based machine translation. In particular, Son et al. (2012) and Schwenk (2012) employed a neural network to model the phrase translation probability on the rule level  $\langle \alpha, \gamma \rangle$  instead of the bilingual sentence level  $\langle f, e \rangle$  as in Eq. (5), and thus they did not go beyond the log-linear model for SMT.

There are also works which exploit non-linear models in SMT. Duh and Kirchhoff (2008) proposed a boosting re-ranking algorithm using MERT as a weak learner to improve the model’s expressive abilities; Sokolov et al. (2012) similarly proposed a boosting re-ranking method from the ranking perspective rather than the classification perspective. Instead of considering the re-ranking task in SMT, Xiao et al. (2010) employed a boosting method for the system combination in SMT. Unlike their post-processing models (either a re-ranking or a system combination model) in SMT, we propose a non-linear translation model which can be easily incorporated into the existing SMT framework.

## 7 Conclusion and Future Work

In this paper, we go beyond the log-linear model for SMT and propose a novel AdNN based trans-

lation model. Our model overcomes some of the shortcomings suffered by the log-linear model: linearity and the lack of deep interpretation and representation in features. One advantage of our model is that it jointly learns features and tunes the translation model and thus learns features towards the translation evaluation metric. Additionally, the decoding of our model is as efficient as that of the log-linear model. For Chinese-to-English and Japanese-to-English translation tasks, our model significantly outperforms the log-linear model, with the help of word embedding.

We plan to explore more work on the additive neural networks in the future. For example, we will train word embedding matrices for source and target languages from a larger corpus, and take into consideration the bilingual information, for instance, word alignment; the multi-layer neural network within the additive neural networks will be also investigated in addition to the single-layer neural network; and we will test our method on other translation tasks with larger training data as well.

## Acknowledgments

We would like to thank our colleagues in both HIT and NICT for insightful discussions, Tomas Mikolov for the helpful discussion about the word embedding in RNNLM, and three anonymous reviewers for many invaluable comments and suggestions to improve our paper. This work is supported by National Natural Science Foundation of China (61173073, 61100093, 61073130, 61272384), the Key Project of the National High Technology Research and Development Program of China (2011AA01A207), and the Fundamental Research Funds for Central Universities (HIT.NSRIF.2013065).

## References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March.

Christopher M. Bishop. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: pa-

rameter estimation. *Comput. Linguist.*, 19:263–311, June.

Andreas Buja, Trevor Hastie, and Robert Tibshirani. 1989. Linear smoothers and additive models. *The Annals of Statistics*, 17:453–510.

M. Asuncin Castaño, Francisco Casacuberta, and Enrique Vidal. 1997. Machine translation using neural networks and finite-state models. In *TMI*, pages 160–167.

David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proc. of EMNLP. ACL*.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05*, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Chiang. 2007. Hierarchical phrase-based translation. *Comput. Linguist.*, 33(2):201–228, June.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2, HLT '11*, pages 176–181, Stroudsburg, PA, USA. Association for Computational Linguistics.

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning, ICML*.

R. Collobert. 2011. Deep learning for efficient discriminative parsing. In *AISTATS*.

Thomas Deselaers, Saša Hasan, Oliver Bender, and Hermann Ney. 2009. A deep learning approach to machine transliteration. In *Proceedings of the Fourth Workshop on Statistical Machine Translation, StatMT '09*, pages 233–241, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kevin Duh and Katrin Kirchhoff. 2008. Beyond log-linear models: Boosted minimum error rate training for n-best re-ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 37–40, Columbus, Ohio, June. Association for Computational Linguistics.

Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, and Takehito Utsuro. 2010. Overview of the patent translation task at the ntcir-8 workshop. In *In Proceedings of the 8th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access*, pages 293–302.

- William W. Hager and Hongchao Zhang. 2006. Algorithm 851: Cg-descent, a conjugate gradient method with guaranteed descent. *ACM Trans. Math. Softw.*, 32(1):113–137, March.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*. ACL.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Philipp Koehn. 2004a. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*.
- Philipp Koehn. 2004b. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP*. ACL.
- Quoc V. Le, Jiquan Ngiam, Adam Coates, Ahbik Lahiri, Bobby Prochnow, and Andrew Y. Ng. 2011. On optimization methods for deep learning. In *ICML*, pages 265–272.
- Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048.
- Patrick Nguyen, Milind Mahajan, and Xiaodong He. 2007. Training non-parametric features for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 72–79, Prague, Czech Republic, June. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 440–447, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 295–302, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.
- William J. E. Potts. 1999. Generalized additive neural networks. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '99, pages 194–200, New York, NY, USA. ACM.
- Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of the 24th International Conference on Computational Linguistics*, COLING '12, Mumbai, India. Association for Computational Linguistics.
- Richard Socher, Cliff Chung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*.
- A. Sokolov, G. Wisniewski, and F. Yvon. 2012. Non-linear n-best list reranking with few features. In *AMTA*, San Diego, USA.
- Le Hai Son, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 39–48, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Andreas Stolcke. 2002. Srlm - an extensible language modeling toolkit. In *Proc. of ICSLP*.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics.
- D. A. de Waal and J. V. du Toit. 2007. Generalized additive models from a neural network perspective. In *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, ICDMW '07, pages 265–270, Washington, DC, USA. IEEE Computer Society.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773, Prague, Czech Republic, June. Association for Computational Linguistics.

Tong Xiao, Jingbo Zhu, Muhua Zhu, and Huizhen Wang. 2010. Boosting-based system combination for machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 739–748, Stroudsburg, PA, USA. Association for Computational Linguistics.

# Hierarchical Phrase Table Combination for Machine Translation

Conghui Zhu<sup>1</sup> Taro Watanabe<sup>2</sup> Eiichiro Sumita<sup>2</sup> Tiejun Zhao<sup>1</sup>

<sup>1</sup>School of Computer Science and Technology

Harbin Institute of Technology (HIT), Harbin, China

<sup>2</sup>National Institute of Information and Communication Technology

3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan

{chzhu,tjzhao}@mtlab.hit.edu.cn

{taro.watanabe,Sumita}@nict.go.jp

## Abstract

Typical statistical machine translation systems are batch trained with a given training data and their performances are largely influenced by the amount of data. With the growth of the available data across different domains, it is computationally demanding to perform batch training every time when new data comes. In face of the problem, we propose an efficient phrase table combination method. In particular, we train a Bayesian phrasal inversion transduction grammars for each domain separately. The learned phrase tables are hierarchically combined as if they are drawn from a hierarchical Pitman-Yor process. The performance measured by BLEU is at least as comparable to the traditional batch training method. Furthermore, each phrase table is trained separately in each domain, and while computational overhead is significantly reduced by training them in parallel.

## 1 Introduction

Statistical machine translation (SMT) systems usually achieve 'crowd-sourced' improvements with batch training. Phrase pair extraction, the key step to discover translation knowledge, heavily relies on the scale of training data. Typically, the more parallel corpora used, the more phrase pairs and more accurate parameters will be learned, which can obviously be beneficial to improving translation performances. Today, more parallel sentences are drawn from divergent domains, and the size keeps growing. Consequently, how to effectively use those data and improve translation performance becomes a challenging issue.

This joint work was done while the first author visited NICT.

Batch retraining is not acceptable for this case, since it demands serious computational overhead when training on a large data set, and it requires us to re-train every time new training data is available. Even if we can handle the large computation cost, improvement is not guaranteed every time we perform batch tuning on the newly updated training data obtained from divergent domains. Traditional domain adaption methods for SMT are also not adequate in this scenario. Most of them have been proposed in order to make translation systems perform better for resource-scarce domains when most training data comes from resource-rich domains, and ignore performance on a more generic domain without domain bias (Wang et al., 2012). As an alternative, incremental learning may resolve the gap by incrementally adding data sentence-by-sentence into the training data. Since SMT systems trend to employ very large scale training data for translation knowledge extraction, updating several sentence pairs each time will be annihilated in the existing corpus.

This paper proposes a new phrase table combination method. First, phrase pairs are extracted from each domain without interfering with other domains. In particular, we employ the non-parametric Bayesian phrasal inversion transduction grammar (ITG) of Neubig et al. (2011) to perform phrase table extraction. Second, extracted phrase tables are combined as if they are drawn from a hierarchical Pitman-Yor process, in which the phrase tables represented as tables in the Chinese restaurant process (CRP) are hierarchically chained by treating each of the previously learned phrase tables as prior to the current one. Thus, we can easily update the chain of phrase tables by appending the newly extracted phrase table and by treating the chain of the previous ones as its prior.

Experiment results indicate that our method can achieve better translation performance when there exists a large divergence in domains, and can

achieve at least comparable results to batch training methods, with a significantly less computational overhead.

The rest of the paper is organized as follows. In Section 2, we introduce related work. In section 3, we briefly describe the translation model with phrasal ITGs and Pitman-Yor process. In section 4, we explain our hierarchical combination approach and give experiment results in section 5. We conclude the paper in the last section.

## 2 Related Work

Bilingual phrases are cornerstones for phrase-based SMT systems (Och and Ney, 2004; Koehn et al., 2003; Chiang, 2005) and existing translation systems often get ‘crowd-sourced’ improvements (Levenberg et al., 2010). A number of approaches have been proposed to make use of the full potential of the available parallel sentences from various domains, such as domain adaptation and incremental learning for SMT.

The translation model and language model are primary components in SMT. Previous work proved successful in the use of large-scale data for language models from diverse domains (Brants et al., 2007; Schwenk and Koehn, 2008). Alternatively, the language model is incrementally updated by using a succinct data structure with a interpolation technique (Levenberg and Osborne, 2009; Levenberg et al., 2011).

In the case of the previous work on translation modeling, mixed methods have been investigated for domain adaptation in SMT by adding domain information as additional labels to the original phrase table (Foster and Kuhn, 2007). Under this framework, the training data is first divided into several parts, and phrase pairs are extracted with some sub-domain features. Then all the phrase pairs and features are tuned together with different weights during decoding. As a way to choose the right domain for the domain adaptation, a classifier-based method and a feature-based method have been proposed. Classification-based methods must at least add an explicit label to indicate which domain the current phrase pair comes from. This is traditionally done with an automatic domain classifier, and each input sentence is classified into its corresponding domain (Xu et al., 2007). As an alternative to the classification-based approach, Wang et al. (2012) employed a feature-based approach, in which phrase pairs are enriched

by a feature set to potentially reflect the domain information. The similarity calculated by a information retrieval system between the training subset and the test set is used as a feature for each parallel sentence (Lu et al., 2007). Monolingual topic information is taken as a new feature for a domain adaptive translation model and tuned on the development set (Su et al., 2012). Regardless of underlying methods, either classifier-based or feature-based method, the performance of current domain adaptive phrase extraction methods is more sensitive to the development set selection. Usually the domain similar to a given development data is usually assigned higher weights.

Incremental learning in which new parallel sentences are incrementally updated to the training data is employed for SMT. Compared to traditional frequent batch oriented methods, an online EM algorithm and active learning are applied to phrase pair extraction and achieves almost comparable translation performance with less computational overhead (Levenberg et al., 2010; González-Rubio et al., 2011). However, their methods usually require numbers of hyperparameters, such as mini-batch size, step size, or human judgment to determine the quality of phrases, and still rely on a heuristic phrase extraction method in each phrase table update.

## 3 Phrase Pair Extraction with Unsupervised Phrasal ITGs

Recently, phrase alignment with ITGs (Cherry and Lin, 2007; Zhang et al., 2008; Blunsom et al., 2008) and parameter estimation with Gibbs sampling (DeNero and Klein, 2008; Blunsom and Cohn, 2010) are popular. Here, we employ a method proposed by Neubig et al. (2011), which uses parametric Bayesian inference with the phrasal ITGs (Wu, 1997). It can achieve comparable translation accuracy with a much smaller phrase table than the traditional GIZA++ and heuristic phrase extraction methods. It has also been proved successful in adjusting the phrase length granularity by applying character-based SMT with more sophisticated inference (Neubig et al., 2012).

ITG is a synchronous grammar formalism which analyzes bilingual text by introducing inverted rules, and each ITG derivation corresponds to the alignment of a sentence pair (Wu, 1997). Translation probabilities of ITG phrasal align-

ments can be estimated in polynomial time by slightly limiting word reordering (DeNero and Klein, 2008).

More formally,  $P(\langle e, f \rangle; \theta_x, \theta_t)$  are the probability of phrase pairs  $\langle e, f \rangle$ , which is parameterized by a phrase pair distribution  $\theta_t$  and a symbol distribution  $\theta_x$ .  $\theta_x$  is a Dirichlet prior, and  $\theta_t$  is estimated with the Pitman-Yor process (Pitman and Yor, 1997; Teh, 2006), which is expressed as

$$\theta_t \sim PY(d, s, P_{dac}) \quad (1)$$

where  $d$  is the discount parameter,  $s$  is the strength parameter, and  $P_{dac}$  is a prior probability which acts as a fallback probability when a phrase pair is not in the model.

Under this model, the probability for a phrase pair found in a bilingual corpus  $\langle E, F \rangle$  can be represented by the following equation using the Chinese restaurant process (Teh, 2006):

$$P(\langle e_i, f_i \rangle; \langle E, F \rangle) = \frac{1}{C + s}(c_i - d \times t_i) + \frac{1}{C + s}(s + d \times T) \times P_{dac}(\langle e_i, f_i \rangle) \quad (2)$$

where

1.  $c_i$  and  $t_i$  are the customer and table count of the  $i$ th phrase pair  $\langle e_i, f_i \rangle$  found in a bilingual corpus  $\langle E, F \rangle$ ;
2.  $C$  and  $T$  are the total customer and table count in corpus  $\langle E, F \rangle$ ;
3.  $d$  and  $s$  are the discount and strengthen hyper-parameters.

The prior probability  $P_{dac}$  is recursively defined by breaking a longer phrase pair into two through the recursive ITG's generative story as follows (Neubig et al., 2011):

1. Generate symbol  $x$  from  $P_x(x; \theta_x)$  with three possible values: *Base*, *REG*, or *INV*.
2. Depending on the value of  $x$  take the following actions.
  - a. If  $x = \textit{Base}$ , generate a new phrase pair directly from  $P_{base}$ .
  - b. If  $x = \textit{REG}$ , generate  $\langle e_1, f_1 \rangle$  and  $\langle e_2, f_2 \rangle$  from  $P(\langle e, f \rangle; \theta_x, \theta_t)$ , and concatenate them into a single phrase pair  $\langle e_1 e_2, f_1 f_2 \rangle$ .

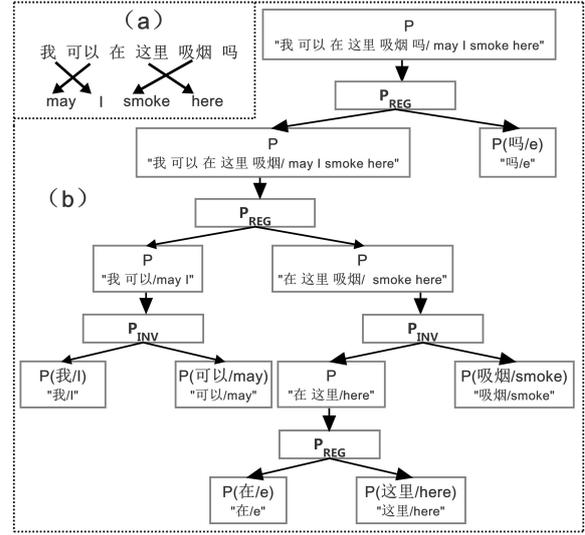


Figure 1: A word alignment (a), and its hierarchical derivation (b).

- c. If  $x = \textit{INV}$ , follow a similar process as b, but concatenate  $f_1$  and  $f_2$  in reverse order  $\langle e_1 e_2, f_2 f_1 \rangle$ .

Note that the  $P_{dac}$  is recursively defined through the binary branched  $P$ , which in turns employs  $P_{dac}$  as a prior probability.  $P_{base}$  is a base measure defined as a combination of the IBM Models in two directions and the unigram language models in both sides. Inference is carried out by a heuristic beam search based block sampling with an efficient look ahead for a faster convergence (Neubig et al., 2012).

Compared to GIZA++ with heuristic phrase extraction, the Bayesian phrasal ITG can achieve competitive accuracy under a smaller phrase table size. Further, the fallback model can incorporate phrases of all granularity by following the ITG's recursive definition. Figure 1 (b) illustrates an example of the phrasal ITG derivation for word alignment in Figure 1 (a) in which a bilingual sentence pair is recursively divided into two through the recursively defined generative story.

## 4 Hierarchical Phrase Table Combination

We propose a new phrase table combination method, in which individually learned phrase table are hierarchically chained through a hierarchical Pitman-Yor process.

Firstly, we assume that the whole training data  $\langle E, F \rangle$  can be split into  $J$  domains,  $\{\langle E^1, F^1 \rangle, \dots, \langle E^J, F^J \rangle\}$ . Then phrase pairs are

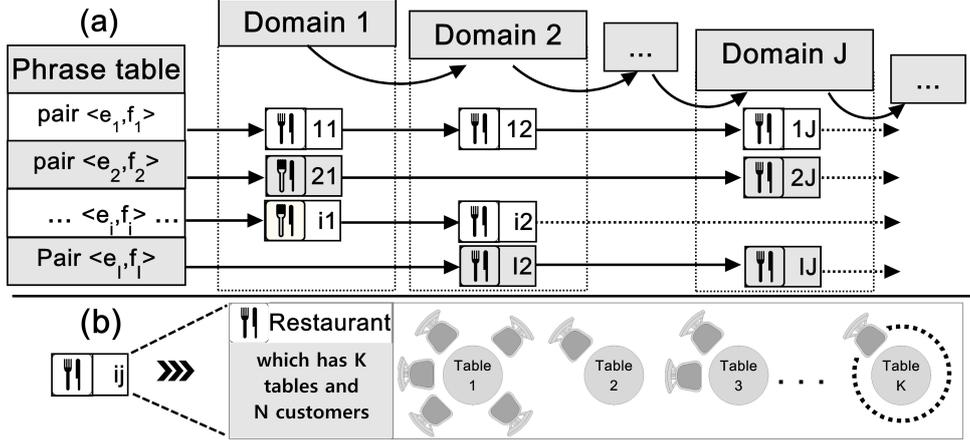


Figure 2: A hierarchical phrase table combination (a), and a basic unit of a Chinese restaurant process with  $K$  tables and  $N$  customers.

extracted from each domain  $j$  ( $1 \leq j \leq J$ ) separately with the method introduced in Section 3. In traditional domain adaptation approaches, phrase pairs are extracted together with their probabilities and/or frequencies so that the extracted phrase pairs are merged uniformly or after scaling.

In this work, we extract the table counts for each phrase pair under the Chinese restaurant process given in Section 3. In Figure 2 (b), a CRP is illustrated which has  $K$  tables and  $N$  customers with each chair representing a customer. Meanwhile there are two parameters, discount and strength for each domain similar to the ones in Equation (1).

Our proposed hierarchical phrase table combination can be formally expressed as following:

$$\begin{aligned}
 \theta^1 &\sim PY(d^1, s^1, P^2) \\
 \dots &\dots \\
 \theta^j &\sim PY(d^j, s^j, P^{j+1}) \\
 \dots &\dots \\
 \theta^J &\sim PY(d^J, s^J, P_{base}^J) \quad (3)
 \end{aligned}$$

Here the  $(j + 1)$ th layer hierarchical Pitman-Yor process is employed as a base measure for the  $j$ th layer hierarchical Pitman-Yor process. The hierarchical chain is terminated by the base measure from the  $J$ th domain  $P_{base}^J$ . The hierarchical structure is illustrated in Figure 2 (a) in which the solid lines implies a fall back using the table counts from the subsequent domains, and the dotted lines means the final fallback to the base measure  $P_{base}^J$ . When we query a probability of a phrase pair  $\langle e, f \rangle$ , we first query the probability of the first layer  $P^1(\langle e, f \rangle)$ . If  $\langle e, f \rangle$  is not in the model, we will fallback to the next level of

$P^2(\langle e, f \rangle)$ . This process continues until we reach the  $J_{th}$  base measure of  $P^J(\langle e, f \rangle)$ . Each fallback can be viewed as a translation knowledge integration process between subsequent domains.

For example in Figure 2 (a), the  $i$ th phrase pair  $\langle e_i, f_i \rangle$  appears only in the domain 1 and domain 2, so its translation probability can be calculated by substituting Equation (3) with Equation (2):

$$\begin{aligned}
 P(\langle e_i, f_i \rangle; \langle E, F \rangle) &= \frac{1}{C^1 + s^1} (c_i^1 - d^1 \times t_i^1) \\
 &+ \frac{s^1 + d^1 \times T^1}{(C^1 + s^1) \times (C^2 + s^2)} (c_i^2 - d^2 \times t_i^2) \\
 &+ \prod_{j=1}^J \left( \frac{s^j + d^j \times T^j}{C^j + s^j} \right) \times P_{base}^J(\langle e_i, f_i \rangle) \quad (4)
 \end{aligned}$$

where the superscript indicates the domain for the corresponding counts, i.e.  $c_i^j$  for the customer count in the  $j$ th domain. The first term in Equation (4) is the phrase probability from the first domain, and the second one comes from the second domain, but weighted by the fallback weight of the 1st domain. Since  $\langle e_i, f_i \rangle$  does not appear in the rest of the layers, the last term is taken from all the fallback weight from the second layer to the  $J$ th layer with the final  $P_{base}^J$ . All the parameters  $\theta^j$  and hyperparameters  $d^j$  and  $s^j$ , are obtained by learning on the  $j$ th domain. Returning the hyperparameters again when cascading another domain may improve the performance of the combination weight, but we will leave it for future work. The hierarchical process can be viewed as an instance of adapted integration of translation knowledge from each sub-domain.

**Algorithm 1 Translation Probabilities Estimation****Input:**  $c_i^j, t_i^j, P_{base}^j, C^j, T^j, d^j$  and  $s^j$ **Output:** The translation probabilities for each pair

```

1: for all phrase pair  $\langle e_i, f_i \rangle$  do
2:   Initialize the  $P(\langle e_i, f_i \rangle) = 0$  and  $w_i = 1$ 
3:   for all domain  $\langle E_j, F_j \rangle$  such that  $1 \leq j \leq J - 1$  do
4:     if  $\langle e_i, f_i \rangle \in \langle E_j, F_j \rangle$  then
5:        $P(\langle e_i, f_i \rangle) += w_i \times (C_i^j - d^j \times t_i^j) / (C^j + s^j)$ 
6:     end if
7:      $w_i = w_i \times (s^j + d^j \times T^j) / (C^j + s^j)$ 
8:   end for
9:    $P(\langle e_i, f_i \rangle) += w_i \times (C_i^J - d^J \times t_i^J + (s^J + d^J \times T^J) \times P_{base}^J(\langle e_i, f_i \rangle)) / (C^J + s^J)$ 
10: end for

```

Our approach has several advantages. First, each phrase pair extraction can concentrate on a small portion of domain-specific data without interfering with other domains. Since no tuning stage is involved in the hierarchical combination, we can easily include a new phrase table from a new domain by simply chaining them together. Second, phrase pair phrase extraction in each domain is completely independent, so it is easy to parallelize in a situation where the training data is too large to fit into a small amount of memory. Finally, new domains can be integrated incrementally. When we encounter a new domain, and if a phrase pair is completely new in terms of the model, the phrase pair is simply appended to the current model, and computed without the fallback probabilities, since otherwise, the phrase pair would be boosted by the fallback probabilities. Pitman-Yor process is also employed in n-gram language models which are hierarchically represented through the hierarchical Pitman-Yor process with switch priors to integrate different domains in all the levels (Wood and Teh, 2009). Our work incrementally combines the models from different domains by directly employing the hierarchical process through the base measures.

## 5 Experiment

We evaluate the proposed approach on the Chinese-to-English translation task with three data sets with different scales.

Data set	Corpus	#sent. pairs
IWSLT	HIT	52,603
	BTEC	19,975
FBIS	Domain 1	47,993
	Domain 2	30,272
	Domain 3	49,509
	Domain 4	38,228
	Domain 5	55,913
LDC	News	221,915
	News	95,593
	Magazine	98,335
	Magazine	254,488
	Finance	86,112

Table 1: The sentence pairs used in each data set.

### 5.1 Experiment Setup

The first data set comes from the IWSLT2012 OLYMPICS task consisting of two training sets: the HIT corpus, which is closely related to the Beijing 2008 Olympic Games, and the BTEC corpus, which is a multilingual speech corpus containing tourism-related sentences. The second data set, the FBIS corpus, is a collection of news articles and does not have domain information itself, so a Latent Dirichlet Allocation (LDA) tool, PLDA<sup>1</sup>, is used to divide the whole corpus into 5 different sub-domains according to the concatenation of the source side and target side as a single sentence (Liu et al., 2011). The third data set is composed of 5 corpora<sup>2</sup> from LDC with various domains, including news, magazine, and finance. The details are shown in Table 1.

In order to evaluate our approach, four phrase pair extraction methods are performed:

1. GIZA-linear: Phrase pairs are extracted in each domain by GIZA++ (Och and Ney, 2003) and the "grow-diag-final-and" method with a maximum length 7. The phrase tables from various domains are linearly combined by averaging the feature values.
2. Pialign-linear: Similar to GIZA-linear, but we employed the phrasal ITG method described in Section 3 using the pialign toolkit<sup>3</sup> (Neubig et

<sup>1</sup><http://code.google.com/p/plda/><sup>2</sup>In particular, they come from LDC catalog number: LDC2002E18, LDC2002E58, LDC2003E14, LDC2005E47, LDC2006E26, in this order.<sup>3</sup><http://www.phontron.com/pialign/>

Methods	IWSLT		FBIS		LDC	
	BLEU	Size	BLEU	Size	BLEU	Size
GIZA-linear	19.222	1,200,877	29.342	15,369,028	30.67	77,927,347
Pialign-linear	19.534	876,059	29.858	7,235,342	31.12	28,877,149
GIZA-batch	19.616	1,185,255	<b>31.38</b>	13,737,258	<b>32.06</b>	63,606,056
Pialign-batch	19.506	841,931	31.104	6,459,200		
Pialign-adaptive	19.624	841,931	30.926	6,459,200		
Hier-combin	<b>20.32</b>	876,059	31.29	7,235,342	32.03	28,877,149

Table 2: BLEU scores and phrase table size by alignment method and probabilities estimation method. Pialign was run with five samples. Because of computational overhead, the baseline Pialign-batch and Pialign-adaptive were not run on the largest data set.

- al., 2011). Extracted phrase pairs are linearly combined by averaging the feature values.
3. GIZA-batch: Instead of splitting into each domain, the data set is merged as a single corpus and then a heuristic GZA-based phrase extraction is performed, similar as GIZA-linear.
  4. Pialign-batch: Similar to the GIZA-batch, a single model is estimated from a single, merged corpus. Since pialign cannot handle large data, we did not experiment on the largest LDC data set.
  5. Pialign-adaptive: Alignment and phrase pairs extraction are same to Pialign-batch, while translation probabilities are estimated by the adaptive method with monolingual topic information (Su et al., 2012). The method established the relationship between the out-of-domain bilingual corpus and in-domain monolingual corpora via topic distribution to estimate the translation probability.

$$\begin{aligned}
\phi(\tilde{e}|\tilde{f}) &= \sum_{t_f} \phi(\tilde{e}, t_f|\tilde{f}) \\
&= \sum_{t_f} \phi(\tilde{e}|t_f, \tilde{f}) \cdot P(t_f|\tilde{f})
\end{aligned} \tag{5}$$

where  $\phi(\tilde{e}|t_f, \tilde{f})$  is the probability of translating  $\tilde{f}$  into  $\tilde{e}$  given the source-side topic  $\tilde{f}$ ,  $P(t_f|\tilde{f})$  is the phrase-topic distribution of  $\tilde{f}$ .

The method we proposed is named Hier-combin. It extracts phrase pairs in the same way as the Pialign-linear. In the phrase table combination process, the translation probability of each phrase pair is estimated by the Hier-combin and the other features are also linearly combined by averaging

the feature values. Pialign is used with default parameters. The parameter 'samps' is set to 5, which indicates 5 samples are generated for a sentence pair.

The IWSLT data consists of roughly 2,000 sentences and 3,000 sentences each from the HIT and BTEC for development purposes, and the test data consists of 1,000 sentences. For the FBIS and LDC task, we used NIST MT 2002 and 2004 for development and testing purposes, consisting of 878 and 1,788 sentences respectively. We employ Moses, an open-source toolkit for our experiment (Koehn et al., 2007). SRILM Toolkit (Stolcke, 2002) is employed to train 4-gram language models on the Xinhua portion of Gigaword corpus, while for the IWSLT2012 data set, only its training set is used. We use batch-MIRA (Cherry and Foster, 2012) to tune the weight for each feature and translation quality is evaluated by the case-insensitive BLEU-4 metric (Papineni et al., 2002). The BLEU scores reported in this paper are the average of 5 independent runs of independent batch-MIRA weight training, as suggested by (Clark et al., 2011).

## 5.2 Result and Analysis

### 5.2.1 Performances of various extraction methods

We carry out a series of experiments to evaluate translation performance. The results are listed in Table 2. Our method significantly outperforms the baseline Pialign-linear. Except for the translation probabilities, the phrase pairs of two methods are exactly same, so the number of phrase pairs are equal in the two methods. Further more, the performance of the baseline Pialign-adaptive is also higher than the baseline Pialign-linear's and lower than ours. This proves that the adaptive method

Methods	Task	Time(minute)
Batch	Retraining	536.9
Hierarchical Combination	Parallel Extraction	122.55
	Integrating	1.5
	Total	124.05

Table 3: Minutes used for alignment and phrase pair extraction in the FBIS data set.

with monolingual topic information is useful in the tasks, but our approach with the hierarchical Pitman-Yor process can estimate more accurate translation probabilities based on all the data from various domains.

Compared with the GIZA-batch, our approach achieves competitive performance with a much smaller phrase table. The number of phrase pairs generated by our method is only 73.9%, 52.7%, and 45.4% of the GIZA-batch’s respectively. In the IWSLT2012 data set, there is a huge difference gap between the HIT corpus and the BTEC corpus, and our method gains 0.814 BLEU improvement. While the FBIS data set is artificially divided and no clear human assigned differences among sub-domains, our method loses 0.09 BLEU.

In the framework we proposed, phrase pairs are extracted from each domain completely independent of each other, so those tasks can be executed on different machines, at different times, and of course in parallel when we assume that the domains are not incrementally added in the training data. The runtime of our approach and the batch-based ITGs sampling method in the FBIS data set is listed in Table 3 measured on a 2.7 GHz E5-2680 CPU and 128 Gigabyte memory. When comparing the hier-combin with the pialign-batch, the BLEU scores are a little higher while the time spent for training is much lower, almost one quarter of the pialign-batch.

Even the performance of the pialign-linear is better than the Baseline GIZA-linear’s, which means that phrase pair extraction with hierarchical phrasal ITGs and sampling is more suitable for domain adaptation tasks than the combination GIZA++ and a heuristic method.

Generally, the hierarchical combination method exploits the nature of a hierarchical Pitman-Yor process and gains the advantage of its smoothing effect, and our approach can incrementally generate a succinct phrase table based on all the data from various domains with more accurate prob-

abilities. Traditional SMT phrase pair extraction is batch-based, while our method has no obvious shortcomings in translation accuracy, not to mention efficiency.

### 5.2.2 Effect of Integration Order

Here, we evaluate whether our hierarchical combination is sensitive to the order of the domains when forming a hierarchical structure. Through Equation (3), in our experiments, we chained the domains in the order listed in Table 1, which is in almost chronological order. Table 4 shows the BLEU scores for the three data sets, in which the order of combining phrase tables from each domain is alternated in the ascending and descending of the similarity to the test data. The similarity between the data from each domain and the test data is calculated using the perplexity measure with 5-gram language model. The model learned from the domain more similar to the test data is placed in the front so that it can largely influence the parameter computation with less backoff effects. There is a big difference between the two opposite order in IWSLT 2012 data set, in which more than one point of decline in BLEU score when taking the BTEC corpus as the first layer. Note that the perplexity of BTEC was 344.589 while that of HIT was 107.788. The result may indicate that our hierarchical phrase combination method is sensitive to the integration order when the training data is small and there exists large gap in the similarity. However, if most domains are similar (FBIS data set) or if there are enough parallel sentence pairs (NIST data set) in each domain, then the translation performances are almost similar even with the opposite integrating orders.

	IWSLT	FBIS	LDC
Descending	20.154	30.491	31.268
Ascending	19.066	30.388	31.254
Difference	<b>1.088</b>	<b>0.103</b>	<b>0.014</b>

Table 4: BLEU scores for the hierarchical model with different integrating orders. Here Pialign was run without multi-samples.

## 6 Conclusion and Future Work

In this paper, we present a novel hierarchical phrase table combination method for SMT, which can exploit more of the potential from all of data coming from various fields and generate a suc-

cinct phrase table with more accurate translation probabilities. The method assumes that a combined model is derived from a hierarchical Pitman-Yor process with each prior learned separately in each domain, and achieves BLEU scores competitive with traditional batch-based ones. Meanwhile, the framework has natural characteristics for parallel and incremental phrase pair extraction. The experiment results on three different data sets indicate the effectiveness of our approach.

In future work, we will also introduce incremental learning for phrase pair extraction inside a domain, which means using the current translation probabilities already obtained as the base measure of sampling parameters for the upcoming domain. Furthermore, we will investigate any tradeoffs between the accuracy of the probability estimation and the coverage of phrase pairs.

## Acknowledgments

We would like to thank our colleagues in both HIT and NICT for insightful discussions, and three anonymous reviewers for many invaluable comments and suggestions to improve our paper. This work is supported by National Natural Science Foundation of China (61100093, 61173073, 61073130, 61272384), and the Key Project of the National High Technology Research and Development Program of China (2011AA01A207).

## References

- Phil Blunsom and Trevor Cohn. 2010. Inducing synchronous grammars with slice sampling. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 238–241, Los Angeles, California, June. Association for Computational Linguistics.
- Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proceedings of ACL*, pages 200–208, Columbus, Ohio, June. Association for Computational Linguistics.
- Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic, June. Association for Computational Linguistics.
- Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montréal, Canada, June. Association for Computational Linguistics.
- Colin Cherry and Dekang Lin. 2007. Inversion transduction grammar for joint phrasal translation modeling. In *Proceedings of SSST, NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 17–24.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05*, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2, HLT '11*, pages 176–181, Stroudsburg, PA, USA. Association for Computational Linguistics.
- John DeNero and Dan Klein. 2008. The complexity of phrase alignment problems. In *Proceedings of ACL-08: HLT, Short Papers*, pages 25–28, Columbus, Ohio, June. Association for Computational Linguistics.
- George Foster and Roland Kuhn. 2007. Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 128–135.
- Jesús González-Rubio, Daniel Ortiz-Martinez, and Francisco Casacuberta. 2011. Fast incremental active learning for statistical machine translation. *AVANCES EN INTELIGENCIA ARTIFICIAL*.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 45–54.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07*, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Abby Levenberg and Miles Osborne. 2009. Stream-based randomised language models for smt. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 756–764. Association for Computational Linguistics.

- Abby Levenberg, Chris Callison-Burch, and Miles Osborne. 2010. Stream-based translation models for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 394–402, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Abby Levenberg, Miles Osborne, and David Matthews. 2011. Multiple-stream language models for statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 177–186, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Zhiyuan Liu, Yuzhou Zhang, Edward Y Chang, and Maosong Sun. 2011. Plda+: Parallel latent dirichlet allocation with data placement and pipeline processing. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–18.
- Yajuan Lu, Jin Huang, and Qun Liu. 2007. Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 343–350, Prague, Czech Republic, June. Association for Computational Linguistics.
- Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 632–641, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Graham Neubig, Taro Watanabe, Shinsuke Mori, and Tatsuya Kawahara. 2012. Machine translation without words through substring alignment. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 165–174, Jeju Island, Korea, July. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.
- Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Comput. Linguist.*, 30(4):417–449, December.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.
- Jim Pitman and Marc Yor. 1997. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25(2):855–900.
- Holger Schwenk and Philipp Koehn. 2008. Large and diverse language models for statistical machine translation. In *International Joint Conference on Natural Language Processing*, pages 661–668.
- Andreas Stolcke. 2002. Srlm - an extensible language modeling toolkit. In *Proc. of ICSLP*.
- Jinsong Su, Hua Wu, Haifeng Wang, Yidong Chen, Xiaodong Shi, Huailin Dong, and Qun Liu. 2012. Translation model adaptation for statistical machine translation with monolingual topic information. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 459–468.
- Yee Whye Teh. 2006. A hierarchical bayesian language model based on pitman-yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 985–992. Association for Computational Linguistics.
- Wei Wang, Klaus Macherey, Wolfgang Macherey, Franz Och, and Peng Xu. 2012. Improved domain adaptation for statistical machine translation. In *Proceedings of the Conference of the Association for Machine translation, Americas*.
- F. Wood and Y. W. Teh. 2009. A hierarchical non-parametric Bayesian approach to statistical language model domain adaptation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 12.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics*, 23(3):377–403.
- Jia Xu, Yonggang Deng, Yuqing Gao, and Hermann Ney. 2007. Domain dependent statistical machine translation. In *Proceedings of the MT Summit XI*.
- Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008. Bayesian learning of non-compositional phrases with synchronous parsing. In *Proceedings of ACL-08: HLT*, pages 97–105, Columbus, Ohio, June. Association for Computational Linguistics.

# Shallow Local Multi Bottom-up Tree Transducers in Statistical Machine Translation

Fabienne Braune and Nina Seemann and Daniel Quernheim and Andreas Maletti

Institute for Natural Language Processing, University of Stuttgart

Pfaffenwaldring 5b, 70569 Stuttgart, Germany

{braunefe, seemanna, daniel, maletti}@ims.uni-stuttgart.de

## Abstract

We present a new translation model integrating the shallow local multi bottom-up tree transducer. We perform a large-scale empirical evaluation of our obtained system, which demonstrates that we significantly beat a realistic tree-to-tree baseline on the WMT 2009 English  $\rightarrow$  German translation task. As an additional contribution we make the developed software and complete tool-chain publicly available for further experimentation.

## 1 Introduction

Besides phrase-based machine translation systems (Koehn et al., 2003), syntax-based systems have become widely used because of their ability to handle non-local reordering. Those systems use synchronous context-free grammars (Chiang, 2007), synchronous tree substitution grammars (Eisner, 2003) or even more powerful formalisms like synchronous tree-sequence substitution grammars (Sun et al., 2009). However, those systems use linguistic syntactic annotation at different levels. For example, the systems proposed by Wu (1997) and Chiang (2007) use no linguistic information and are syntactic in a structural sense only. Huang et al. (2006) and Liu et al. (2006) use syntactic annotations on the source language side and show significant improvements in translation quality. Using syntax exclusively on the target language side has also been successfully tried by Galley et al. (2004) and Galley et al. (2006). Nowadays, open-source toolkits such as Moses (Koehn et al., 2007) offer syntax-based components (Hoang et al., 2009), which allow experiments without expert knowledge. The improvements observed for systems using syntactic annotation on either the source or the target language side naturally led to experiments with models that use syntactic annotations on *both* sides.

However, as noted by Lavie et al. (2008), Liu et al. (2009), and Chiang (2010), the integration of syntactic information on both sides tends to decrease translation quality because the systems become too restrictive. Several strategies such as (i) using parse forests instead of single parses (Liu et al., 2009) or (ii) soft syntactic constraints (Chiang, 2010) have been developed to alleviate this problem. Another successful approach has been to switch to more powerful formalisms, which allow the extraction of more general rules. A particularly powerful model is the non-contiguous version of synchronous tree-sequence substitution grammars (STSSG) of Zhang et al. (2008a), Zhang et al. (2008b), and Sun et al. (2009), which allows sequences of trees on both sides of the rules [see also (Raoult, 1997)]. The multi bottom-up tree transducer (MBOT) of Arnold and Dauchet (1982) and Lilin (1978) offers a middle ground between traditional syntax-based models and STSSG. Roughly speaking, an MBOT is an STSSG, in which all the discontinuities must occur on the target language side (Maletti, 2011). This restriction yields many algorithmic advantages over both the traditional models as well as STSSG as demonstrated by Maletti (2010). Formally, they are expressive enough to express all sensible translations (Maletti, 2012)<sup>1</sup>. Figure 2 displays sample rules of the MBOT variant, called  $\ell$ MBOT, that we use (in a graphical representation of the trees and the alignment).

In this contribution, we report on our novel statistical machine translation system that uses an  $\ell$ MBOT-based translation model. The theoretical foundations of  $\ell$ MBOT and their integration into our translation model are presented in Sections 2 and 3. In order to empirically evaluate the  $\ell$ MBOT model, we implemented a machine trans-

<sup>1</sup>A translation is sensible if it is of linear size increase and can be computed by some (potentially copying) top-down tree transducer.

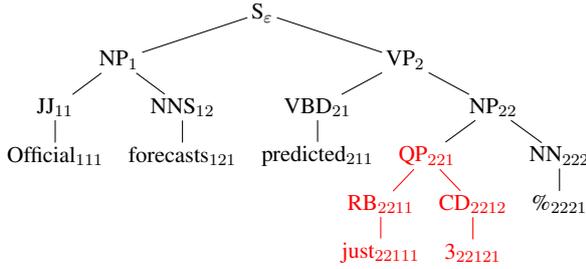


Figure 1: Example tree  $t$  with indicated positions. We have  $t(21) = \text{VBD}$  and  $t|_{221}$  is the subtree marked in red.

lation system that we are going to make available to the public. We implemented  $\ell\text{MBOT}$  inside the syntax-based component of the Moses open source toolkit. Section 4 presents the most important algorithms of our  $\ell\text{MBOT}$  decoder. We evaluate our new system on the WMT 2009 shared translation task English  $\rightarrow$  German. The translation quality is automatically measured using BLEU scores, and we confirm the findings by providing linguistic evidence (see Section 5). Note that in contrast to several previous approaches, we perform large scale experiments by training systems with approx. 1.5 million parallel sentences.

## 2 Theoretical Model

In this section, we present the theoretical generative model used in our approach to syntax-based machine translation. Essentially, it is the local multi bottom-up tree transducer of Maletti (2011) with the restriction that all rules must be shallow, which means that the left-hand side of each rule has height at most 2 (see Figure 2 for shallow rules and Figure 4 for rules including non-shallow rules). The rules extracted from the training example of Figure 3 are displayed in Figure 4. Those extracted rules are forcibly made shallow by removing internal nodes. The application of those rules is illustrated in Figures 5 and 6.

For those that want to understand the inner workings, we recall the principal model in full detail in the rest of this section. Since we utilize syntactic parse trees, let us introduce trees first. Given an alphabet  $\Sigma$  of labels, the set  $T_\Sigma$  of all  $\Sigma$ -trees is the smallest set  $T$  such that  $\sigma(t_1, \dots, t_k) \in T$  for all  $\sigma \in \Sigma$ , integer  $k \geq 0$ , and  $t_1, \dots, t_k \in T$ . Intuitively, a tree  $t$  consists of a labeled root node  $\sigma$  followed by a sequence  $t_1, \dots, t_k$  of its children. A tree  $t \in T_\Sigma$  is *shallow* if  $t = \sigma(t_1, \dots, t_k)$  with  $\sigma \in \Sigma$  and  $t_1, \dots, t_k \in \Sigma$ .

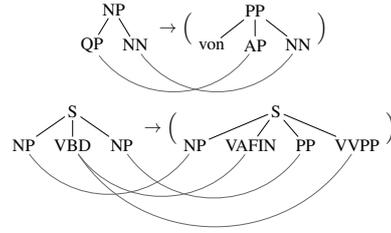


Figure 2: Sample  $\ell\text{MBOT}$  rules.

To address a node inside a tree, we use its position, which is a word consisting of positive integers. Roughly speaking, the root of a tree is addressed with the position  $\varepsilon$  (the empty word). The position  $iw$  with  $i \in \mathbb{N}$  addresses the position  $w$  in the  $i^{\text{th}}$  direct child of the root. In this way, each node in the tree is assigned a unique position. We illustrate this notion in Figure 1. Formally, the *positions*  $\text{pos}(t) \subseteq \mathbb{N}^*$  of a tree  $t = \sigma(t_1, \dots, t_k)$  are inductively defined by  $\text{pos}(t) = \{\varepsilon\} \cup \text{pos}^{(k)}(t_1, \dots, t_k)$ , where

$$\text{pos}^{(k)}(t_1, \dots, t_k) = \bigcup_{1 \leq i \leq k} \{iw \mid w \in \text{pos}(t_i)\} .$$

Let  $t \in T_\Sigma$  and  $w \in \text{pos}(t)$ . The label of  $t$  at position  $w$  is  $t(w)$ , and the subtree rooted at position  $w$  is  $t|_w$ . These notions are also illustrated in Figure 1. A position  $w \in \text{pos}(t)$  is a *leaf* (in  $t$ ) if  $w1 \notin \text{pos}(t)$ . In other words, leaves do not have any children. Given a subset  $N \subseteq \Sigma$ , we let

$$\text{leaf}_N(t) = \{w \in \text{pos}(t) \mid t(w) \in N, w \text{ leaf in } t\}$$

be the set of all leaves labeled by elements of  $N$ . When  $N$  is the set of nonterminals, we call them *leaf nonterminals*. We extend this notion to sequences  $t_1, \dots, t_k \in T_\Sigma$  by

$$\text{leaf}_N^{(k)}(t_1, \dots, t_k) = \bigcup_{1 \leq i \leq k} \{iw \mid w \in \text{leaf}_N(t_i)\} .$$

Let  $w_1, \dots, w_n \in \text{pos}(t)$  be (pairwise prefix-incomparable) positions and  $t_1, \dots, t_n \in T_\Sigma$ . Then  $t[w_i \leftarrow t_i]_{1 \leq i \leq n}$  denotes the tree that is obtained from  $t$  by replacing (in parallel) the subtrees at  $w_i$  by  $t_i$  for every  $1 \leq i \leq n$ .

Now we are ready to introduce our model, which is a minor variation of the local multi bottom-up tree transducer of Maletti (2011). Let  $\Sigma$  and  $\Delta$  be the input and output symbols, respectively, and let  $N \subseteq \Sigma \cup \Delta$  be the set of nonterminal symbols. Essentially, the model works on pairs  $\langle t, (u_1, \dots, u_k) \rangle$  consisting of an input tree  $t \in T_\Sigma$

and a sequence  $u_1, \dots, u_k \in T_\Delta$  of output trees. Such pairs are *pre-translations* of rank  $k$ . The pre-translation  $\langle t, (u_1, \dots, u_k) \rangle$  is *shallow* if all trees  $t, u_1, \dots, u_k$  in it are shallow.

Together with a pre-translation we typically have to store an alignment. Given a pre-translation  $\langle t, (u_1, \dots, u_k) \rangle$  of rank  $k$  and  $1 \leq i \leq k$ , we call  $u_i$  the  $i^{\text{th}}$  translation of  $t$ . An *alignment* for this pre-translation is an injective mapping  $\psi: \text{leaf}_N^{(k)}(u_1, \dots, u_k) \rightarrow \text{leaf}_N(t) \times \mathbb{N}$  such that if  $(w, j) \in \text{ran}(\psi)$ , then also  $(w, i) \in \text{ran}(\psi)$  for all  $1 \leq j \leq i$ .<sup>2</sup> In other words, if an alignment requests the  $i^{\text{th}}$  translation, then it should also request all previous translations.

**Definition 1** A shallow local multi bottom-up tree transducer (*lMBOT*) is a finite set  $R$  of rules together with a mapping  $c: R \rightarrow \mathbb{R}$  such that every rule, written  $t \rightarrow_\psi (u_1, \dots, u_k)$ , contains a shallow pre-translation  $\langle t, (u_1, \dots, u_k) \rangle$  and an alignment  $\psi$  for it.

The components  $t, (u_1, \dots, u_k), \psi$ , and  $c(\rho)$  are called the *left-hand side*, the *right-hand side*, the *alignment*, and the *weight* of the rule  $\rho = t \rightarrow_\psi (u_1, \dots, u_k)$ . Figure 2 shows two example *lMBOT* rules (without weights). Overall, the rules of an *lMBOT* are similar to the rules of an SCFG (synchronous context-free grammar), but our right-hand sides contain a sequence of trees instead of just a single tree. In addition, the alignments in an SCFG rule are bijective between leaf nonterminals, whereas our model permits multiple alignments to a single leaf nonterminal in the left-hand side (see Figure 2).

Our *lMBOT* rules are obtained automatically from data like that in Figure 3. Thus, we (word) align the bilingual text and parse it in both the source and the target language. In this manner we obtain sentence pairs like the one shown in Figure 3. To these sentence pairs we apply the rule extraction method of Maletti (2011). The rules extracted from the sentence pair of Figure 3 are shown in Figure 4. Note that these rules are not necessarily shallow (the last two rules are not). Thus, we post-process the extracted rules and make them shallow. The shallow rules corresponding to the non-shallow rules of Figure 4 are shown in Figure 2.

Next, we define how to combine rules to form derivations. In contrast to most other models, we

<sup>2</sup> $\text{ran}(f)$  for a mapping  $f: A \rightarrow B$  denotes the range of  $f$ , which is  $\{f(a) \mid a \in A\}$ .

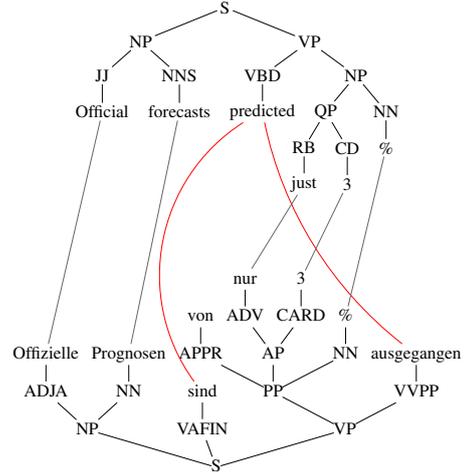


Figure 3: Aligned parsed sentences.

only introduce a derivation semantics that does not collapse multiple derivations for the same input-output pair.<sup>3</sup> We need one final notion. Let  $\rho = t \rightarrow_\psi (u_1, \dots, u_k)$  be a rule and  $w \in \text{leaf}_N(t)$  be a leaf nonterminal (occurrence) in the left-hand side. The  $w$ -rank  $\text{rk}(\rho, w)$  of the rule  $\rho$  is

$$\text{rk}(\rho, w) = \max \{i \in \mathbb{N} \mid (w, i) \in \text{ran}(\psi)\} .$$

For example, for the lower rule  $\rho$  in Figure 2 we have  $\text{rk}(\rho, 1) = 1$ ,  $\text{rk}(\rho, 2) = 2$ , and  $\text{rk}(\rho, 3) = 1$ .

**Definition 2** The set  $\tau(R, c)$  of weighted pre-translations of an *lMBOT*  $(R, c)$  is the smallest set  $T$  subject to the following restriction: If there exist

- a rule  $\rho = t \rightarrow_\psi (u_1, \dots, u_k) \in R$ ,
- a weighted pre-translation

$$\langle t_w, c_w, (u_1^w, \dots, u_k^w) \rangle \in T$$

for every  $w \in \text{leaf}_N(t)$  with

- $\text{rk}(\rho, w) = k_w$ ,<sup>4</sup>
- $t(w) = t_w(\varepsilon)$ ,<sup>5</sup> and
- for every  $iw' \in \text{leaf}_N^{(k)}(u_1, \dots, u_k)$ ,<sup>6</sup>

$$u_i(w') = u_j^v(\varepsilon) \text{ with } \psi(iw') = (v, j),$$

then  $\langle t', c', (u'_1, \dots, u'_k) \rangle \in T$  is a weighted pre-translation, where

- $t' = t[w \leftarrow t_w \mid w \in \text{leaf}_N(t)]$ ,

<sup>3</sup>A standard semantics is presented, for example, in (Maletti, 2011).

<sup>4</sup>If  $w$  has  $n$  alignments, then the pre-translation selected for it has to have suitably many output trees.

<sup>5</sup>The labels have to coincide for the input tree.

<sup>6</sup>Also the labels for the output trees have to coincide.

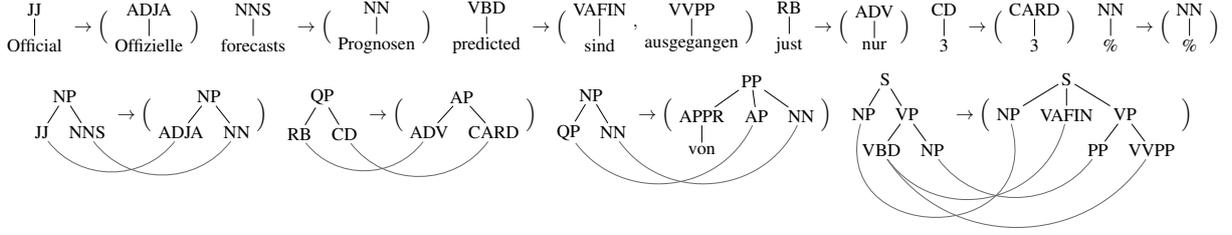


Figure 4: Extracted (even non-shallow) rules. We obtain our rules by making those rules shallow.

- $c' = c(\rho) \cdot \prod_{w \in \text{leaf}_N(t)} c_w$ , and
- $u'_i = u_i[iw' \leftarrow u'_j \mid \psi(iw') = (v, j)]$  for every  $1 \leq i \leq k$ .

Rules that do not contain any nonterminal leaves are automatically weighted pre-translations with their associated rule weight. Otherwise, each nonterminal leaf  $w$  in the left-hand side of a rule  $\rho$  must be replaced by the input tree  $t_w$  of a pre-translation  $\langle t_w, c_w, (u_1^w, \dots, u_{k_w}^w) \rangle$ , whose root is labeled by the same nonterminal. In addition, the rank  $\text{rk}(\rho, w)$  of the replaced nonterminal should match the number  $k_w$  of components in the selected weighted pre-translation. Finally, the nonterminals in the right-hand side that are aligned to  $w$  should be replaced by the translation that the alignment requests, provided that the nonterminal matches with the root symbol of the requested translation. The weight of the new pre-translation is obtained simply by multiplying the rule weight and the weights of the selected weighted pre-translations. The overall process is illustrated in Figures 5 and 6.

### 3 Translation Model

Given a source language sentence  $e$ , our translation model aims to find the best corresponding target language translation  $\hat{g}$ ;<sup>7</sup> i.e.,

$$\hat{g} = \arg \max_g p(g|e) .$$

We estimate the probability  $p(g|e)$  through a log-linear combination of component models with parameters  $\lambda_m$  scored on the pre-translations  $\langle t, (u) \rangle$  such that the leaves of  $t$  concatenated read  $e$ .<sup>8</sup>

$$p(g|e) \propto \prod_{m=1}^7 h_m(\langle t, (u) \rangle)^{\lambda_m}$$

Our model uses the following features  $h_m(\langle t, (u_1, \dots, u_k) \rangle)$  for a general pre-translation  $\tau = \langle t, (u_1, \dots, u_k) \rangle$ :

<sup>7</sup>Our main translation direction is English to German.

<sup>8</sup>Actually,  $t$  must embed in the parse tree of  $e$ ; see Section 4.

- (1) The forward translation weight using the rule weights as described in Section 2
- (2) The indirect translation weight using the rule weights as described in Section 2
- (3) Lexical translation weight source  $\rightarrow$  target
- (4) Lexical translation weight target  $\rightarrow$  source
- (5) Target side language model
- (6) Number of words in the target sentences
- (7) Number of rules used in the pre-translation
- (8) Number of target side sequences; here  $k$  times the number of sequences used in the pre-translations that constructed  $\tau$  (gap penalty)

The rule weights required for (1) are relative frequencies normalized over all rules with the same left-hand side. In the same fashion the rule weights required for (2) are relative frequencies normalized over all rules with the same right-hand side. Additionally, rules that were extracted at most 10 times are discounted by multiplying the rule weight by  $10^{-2}$ . The lexical weights for (2) and (3) are obtained by multiplying the word translations  $w(g_i|e_j)$  [respectively,  $w(e_j|g_i)$ ] of lexically aligned words  $(g_i, e_j)$  across (possibly discontinuous) target side sequences.<sup>9</sup> Whenever a source word  $e_j$  is aligned to multiple target words, we average over the word translations.<sup>10</sup>

$$\begin{aligned} & h_3(\langle t, (u_1, \dots, u_k) \rangle) \\ &= \prod_{\substack{\text{lexical item} \\ e \text{ occurs in } t}} \text{average} \{w(g|e) \mid g \text{ aligned to } e\} \end{aligned}$$

The computation of the language model estimates for (6) is adapted to score partial translations consisting of discontinuous units. We explain the details in Section 4. Finally, the count  $c$  of target sequences obtained in (7) is actually used as a score  $100^{1-c}$ . This discourages rules with many target sequences.

<sup>9</sup>The lexical alignments are different from the alignments used with a pre-translation.

<sup>10</sup>If the word  $e_j$  has no alignment to a target word, then it is assumed to be aligned to a special NULL word and this alignment is scored.

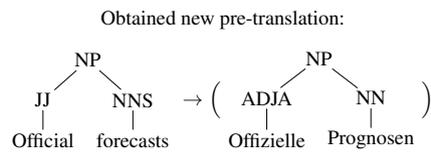
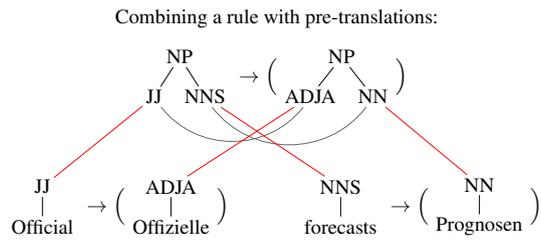


Figure 5: Simple rule application.

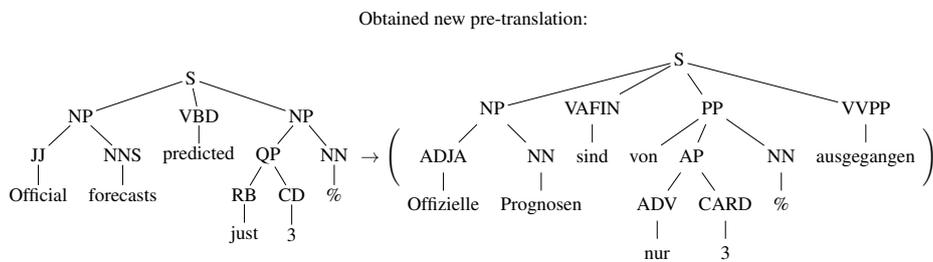
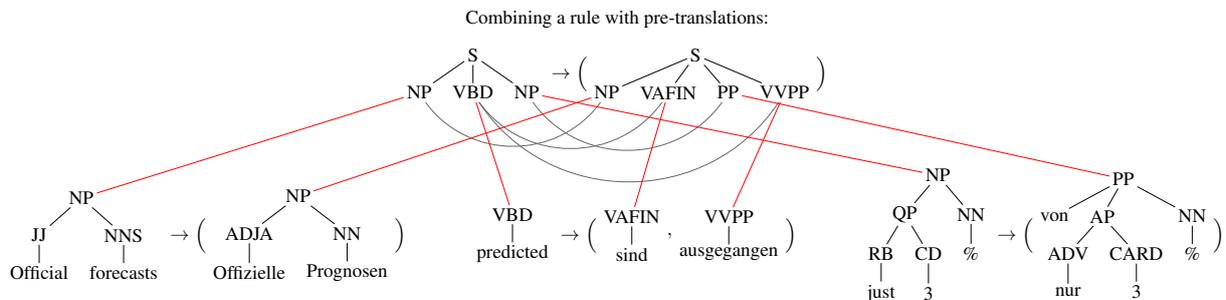


Figure 6: Complex rule application.

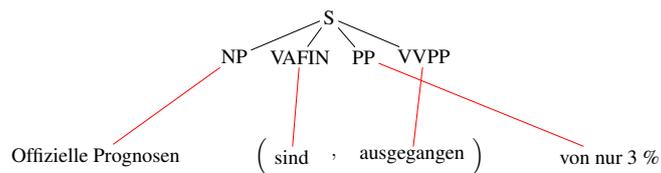


Figure 7: Illustration of LM scoring.

## 4 Decoding

We implemented our model in the syntax-based component of the Moses open-source toolkit by Koehn et al. (2007) and Hoang et al. (2009). The standard Moses syntax-based decoder only handles SCFG rules; i.e., rules with contiguous components on the source and the target language side. Roughly speaking, SCFG rules are  $\ell$ MBOT rules with exactly one output tree. We thus had to extend the system to support our  $\ell$ MBOT rules, in which arbitrarily many output trees are allowed.

The standard Moses syntax-based decoder uses a CYK+ chart parsing algorithm, in which each source sentence is parsed and contiguous spans are processed in a bottom-up fashion. A rule is applicable<sup>11</sup> if the left-hand side of it matches the non-terminal assigned to the full span by the parser and the (non-)terminal assigned to each subspan.<sup>12</sup> In order to speed up the decoding, cube pruning (Chiang, 2007) is applied to each chart cell in order to select the most likely hypotheses for subspans. The language model (LM) scoring is directly integrated into the cube pruning algorithm. Thus, LM estimates are available for all considered hypotheses. To accommodate  $\ell$ MBOT rules, we had to modify the Moses syntax-based decoder in several ways. First, the rule representation itself is adjusted to allow *sequences* of shallow output trees on the target side. Naturally, we also had to adjust hypothesis expansion and, most importantly, language model scoring inside the cube pruning algorithm. An overview of the modified pruning procedure is given in Algorithm 1.

The most important modifications are hidden in lines 5 and 8. The expansion in Line 5 involves matching all nonterminal leaves in the rule as defined in Definition 2, which includes matching all leaf nonterminals in all (discontiguous) output trees. Because the output trees can remain discontiguous after hypothesis creation, LM scoring has to be done individually over all output trees. Algorithm 2 describes our LM scoring in detail. In it we use  $k$  strings  $w_1, \dots, w_k$  to collect the lexical information from the  $k$  output com-

<sup>11</sup>Note that our notion of applicable rules differs from the default in Moses.

<sup>12</sup>Theoretically, this allows that the decoder ignores unary parser nonterminals, which could also disappear when we make our rules shallow; e.g., the parse tree left in the pre-translation of Figure 5 can be matched by a rule with left-hand side NP(Official, forecasts).

---

### Algorithm 1 Cube pruning with $\ell$ MBOT rules

---

#### Data structures:

- $r[i, j]$ : list of rules matching span  $e[i \dots j]$
  - $h[i, j]$ : hypotheses covering span  $e[i \dots j]$
  - $c[i, j]$ : cube of hypotheses covering span  $e[i \dots j]$
- 
- 1: **for all**  $\ell$ MBOT rules  $\rho$  covering span  $e[i \dots j]$  **do**
  - 2:   Insert  $\rho$  into  $r[i, j]$
  - 3:   Sort  $r[i, j]$
  - 4:   **for all**  $(l \rightarrow_{\psi} r) \in r[i, j]$  **do**
  - 5:     Create  $h[i, j]$  by expanding all nonterminals in  $l$  with best scoring hypotheses for subspans
  - 6:     Add  $h[i, j]$  to  $c[i, j]$
  - 7:     **for all** hypotheses  $h \in c[i, j]$  **do**
  - 8:       *Estimate LM score for  $h$*            // see Algorithm 2
  - 9:       Estimate remaining feature scores
  - 10:   Sort  $c[i, j]$
  - 11:   Retrieve first  $\alpha$  elements from  $c[i, j]$  // we use  $\alpha = 10^3$
- 

ponents  $(u_1, \dots, u_k)$  of a rule. These strings can later be rearranged in any order, so we LM-score all of them separately. Roughly speaking, we obtain  $w_i$  by traversing  $u_i$  depth-first left-to-right. If we meet a lexical element (terminal), then we add it to the end of  $w_i$ . On the other hand, if we meet a nonterminal, then we have to consult the best pre-translation  $\tau' = \langle t', (u'_1, \dots, u'_{k'}) \rangle$ , which will contribute the subtree at this position. Suppose that  $u'_j$  will be substituted into the nonterminal in question. Then we first LM-score the pre-translation  $\tau'$  to obtain the string  $w'_j$  corresponding to  $u'_j$ . This string  $w'_j$  is then appended to  $w_i$ . Once all the strings are built, we score them using our 4-gram LM. The overall LM score for the pre-translation is obtained by multiplying the scores for  $w_1, \dots, w_k$ . Clearly, this treats  $w_1, \dots, w_k$  as  $k$  separate strings, although they eventually will be combined into a single string. Whenever such a concatenation happens, our LM scoring will automatically compute  $n$ -gram LM scores based on the concatenation, which in particular means that the LM scores get more accurate for larger spans. Finally, in the final rule only one component is allowed, which yields that the LM indeed scores the complete output sentence.

Figure 7 illustrates our LM scoring for a pre-translation involving a rule with two (discontiguous) target sequences (the construction of the pre-translation is illustrated in Figure 6). When processing the rule rooted at S, an LM estimate is computed by expanding all nonterminal leaves. In our case, these are NP, VAFIN, PP, and VVPP. However, the nodes VAFIN and VVPP are assembled from a (discontiguous) tree sequence. This means that those units have been considered as in-

---

**Algorithm 2** LM scoring

---

**Data structures:**

- $(u_1, \dots, u_k)$ : right-hand side of a rule
  - $(w_1, \dots, w_k)$ :  $k$  strings all initially empty
- 

```
1: score = 1
2: for all  $1 \leq i \leq k$  do
3:   for all leaves  $\ell$  in  $u_i$  (in lexicographic order) do
4:     if  $\ell$  is a terminal then
5:       Append  $\ell$  to  $w_i$ 
6:     else
7:       LM score the best hypothesis for the subspan
8:       Expand  $w_i$  by the corresponding  $w'_j$ 
9:   score = score  $\cdot$  LM( $w_i$ )
```

---

dependent until now. So far, the LM scorer could only score their associated unigrams. However, we also have their associated strings  $w'_1$  and  $w'_2$ , which can now be used. Since VAFIN and VVPP now become parts of a single tree, we can perform LM scoring normally. Assembling the string we obtain

*Offizielle Prognosen sind von nur 3 %  
ausgegangen*

which is scored by the LM. Thus, we first score the 4-grams “Offizielle Prognosen sind von”, then “Prognosen sind von nur”, etc.

## 5 Experiments

### 5.1 Setup

The baseline system for our experiments is the syntax-based component of the Moses open-source toolkit of Koehn et al. (2007) and Hoang et al. (2009). We use linguistic syntactic annotation on both the source and the target language side (tree-to-tree). Our contrastive system is the  $\ell$ MBOT-based translation system presented here. We provide the system with a set of SCFG as well as  $\ell$ MBOT rules. We do not impose any maximal span restriction on either system.

The compared systems are evaluated on the English-to-German<sup>13</sup> news translation task of WMT 2009 (Callison-Burch et al., 2009). For both systems, the used training data is from the 4th version of the *Europarl Corpus* (Koehn, 2005) and the *News Commentary* corpus. Both translation models were trained with approximately 1.5 million bilingual sentences after length-ratio filtering. The word alignments were generated by GIZA++ (Och and Ney, 2003) with the grow-diag-final-and heuristic (Koehn et al., 2005). The

---

<sup>13</sup>Note that our  $\ell$ MBOT-based system can be applied to any language pair as it involves no language-specific engineering.

System	BLEU
<b>Baseline</b>	12.60
$\ell$ <b>MBOT</b>	*13.06
Moses t-to-s	12.72

Table 1: Evaluation results. The starred results are statistically significant improvements over the Baseline (at confidence  $p < 0.05$ ).

English side of the bilingual data was parsed using the Charniak parser of Charniak and Johnson (2005), and the German side was parsed using BitPar (Schmid, 2004) without the function and morphological annotations. Our German 4-gram language model was trained on the German sentences in the training data augmented by the Stuttgart SdeWaC corpus (Web-as-Corpus Consortium, 2008), whose generation is detailed in (Baroni et al., 2009). The weights  $\lambda_m$  in the log-linear model were trained using minimum error rate training (Och, 2003) with the News 2009 development set. Both systems use glue-rules, which allow them to concatenate partial translations without performing any reordering.

### 5.2 Results

We measured the overall translation quality with the help of 4-gram BLEU (Papineni et al., 2002), which was computed on tokenized and lower-cased data for both systems. The results of our evaluation are reported in Table 1. For comparison, we also report the results obtained by a system that utilizes parses only on the source side (Moses tree-to-string) with its standard features.

We can observe from Table 1 that our  $\ell$ MBOT-based system outperforms the baseline. We obtain a BLEU score of 13.06, which is a gain of 0.46 BLEU points over the baseline. This improvement is statistically significant at confidence  $p < 0.05$ , which we computed using the pairwise bootstrap resampling technique of Koehn (2004). Our system is also better than the Moses tree-to-string system. However this improvement (0.34) is not statistically significant. In the next section, we confirm the result of the automatic evaluation through a manual examination of some translations generated by our system and the baseline.

In Table 2, we report the number of  $\ell$ MBOT rules used by our system when decoding the test set. By *lex* we denote rules containing only lexical

	lex	non-term	total
contiguous	23,175	18,355	41,530
discontiguous	315	2,516	2,831

Table 2: Number of rules used in decoding test (lex: only lexical items; non-term: at least one nonterminal).

2-dis	3-dis	4-dis
2,480	323	28

Table 3: Number of  $k$ -discontiguous rules.

items. The label *non-term* stands for rules containing at least one leaf nonterminal. The results show that approx. 6% of all rules used by our  $\ell$ MBOT-system have discontiguous target sides. Furthermore, the reported numbers show that the system also uses rules in which lexical items are combined with nonterminals. Finally, Table 3 presents the number of rules with  $k$  target side components used during decoding.

### 5.3 Linguistic Analysis

In this section we present linguistic evidence supporting the fact that the  $\ell$ MBOT-based system significantly outperforms the baseline. All examples are taken from the translation of the test set used for automatic evaluation. We show that when our system generates better translations, this is directly related to the use of  $\ell$ MBOT rules.

Figures 8 and 9 show the ability of our system to correctly reorder multiple segments in the source sentence where the baseline translates those segments sequentially. An analysis of the generated derivations shows that our system produces the correct translation by taking advantage of rules with discontiguous units on target language side. The rules used in the presented derivations are displayed in Figures 10 and 11. In the first example (Figure 8), we begin by translating “((smuggle)<sub>VB</sub> (eight projectiles)<sub>NP</sub> (into the kingdom)<sub>PP</sub>)<sub>VP</sub>” into the discontiguous sequence composed of (i) “(acht geschosse)<sub>NP</sub>”; (ii) “(in das königreich)<sub>PP</sub>” and (iii) “(schmuggeln)<sub>VP</sub>”. In a second step we assemble all sequences in a rule with contiguous target language side and, at the same time, insert the word “(zu)<sub>PTKZU</sub>” between “(in das königreich)<sub>PP</sub>” and “(schmuggeln)<sub>VP</sub>”.

The second example (Figure 9) illustrates a more complex reordering. First, we trans-

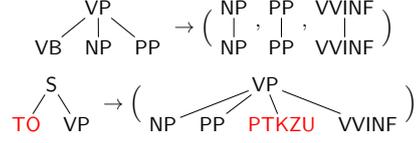


Figure 10: Used  $\ell$ MBOT rules for verbal reordering

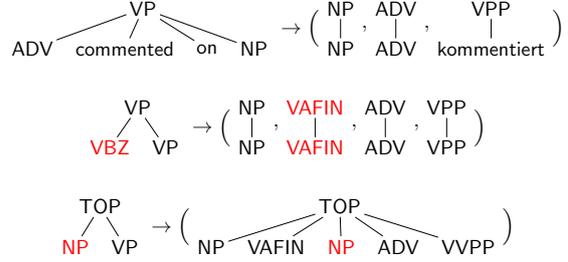


Figure 11: Used  $\ell$ MBOT rules for verbal reordering

late “((again)<sub>ADV</sub> commented on (the problem of global warming)<sub>NP</sub>)<sub>VP</sub>” into the discontiguous sequence composed of (i) “(das problem der globalen erwärmung)<sub>NP</sub>”; (ii) “(wieder)<sub>ADV</sub>” and (iii) “(kommentiert)<sub>VPP</sub>”. In a second step, we translate the auxiliary “(has)<sub>VBZ</sub>” by inserting “(hat)<sub>VAFIN</sub>” into the sequence. We thus obtain, for the input segment “((has)<sub>VBZ</sub> (again)<sub>ADV</sub> commented on (the problem of global warming)<sub>NP</sub>)<sub>VP</sub>”, the sequence (i) “(das problem der globalen erwärmung)<sub>NP</sub>”; (ii) “(hat)<sub>VAFIN</sub>”; (iii) “(wieder)<sub>ADV</sub>”; (iv) “(kommentiert)<sub>VVPP</sub>”. In a last step, the constituent “(president václav klaus)<sub>NP</sub>” is inserted between the discontiguous units “(hat)<sub>VAFIN</sub>” and “(wieder)<sub>ADV</sub>” to form the contiguous sequence “((das problem der globalen erwärmung)<sub>NP</sub> (hat)<sub>VAFIN</sub> (präsident václav klaus)<sub>NP</sub> (wieder)<sub>ADV</sub> (kommentiert)<sub>VVPP</sub>)<sub>TOP</sub>”.

Figures 12 and 13 show examples where our system generates complex words in the target language out of a simple source language word. Again, an analysis of the generated derivation shows that  $\ell$ MBOT takes advantage of rules having several target side components. Examples of such rules are given in Figure 14. Through its ability to use these discontiguous rules, our system correctly translates into reflexive or particle verbs such as “konzentriert sich” (for the English “focuses”) or “besteht darauf” (for the English “insist”). Another phenomenon well handled by our system are relative pronouns. Pronouns such as “that” or “whose” are systematically translated

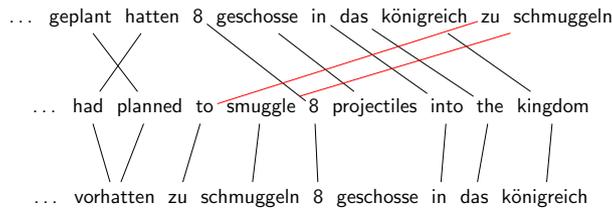


Figure 8: Verbal Reordering (top: our system, bottom: baseline)

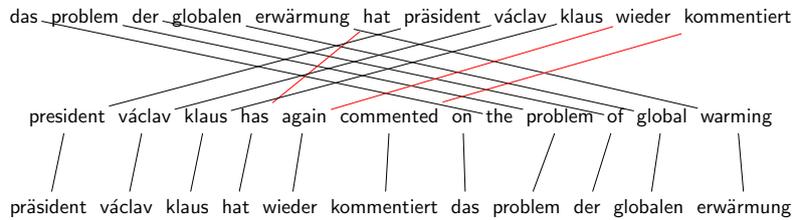


Figure 9: Verbal Reordering (top: our system, bottom: baseline)

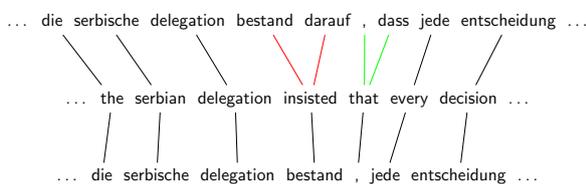


Figure 12: Relative Clause (top: our system, bottom: baseline)



Figure 13: Reflexive Pronoun (top: our system, bottom: baseline)

into both both, “,” and “*dass*” or “,” and “*deren*” (Figure 12).

## 6 Conclusion and Future Work

We demonstrated that our  $\ell$ MBOT-based machine translation system beats a standard tree-to-tree system (Moses tree-to-tree) on the WMT 2009 translation task English  $\rightarrow$  German. To achieve this we implemented the formal model as described in Section 2 inside the Moses machine translation toolkit. Several modifications were necessary to obtain a working system. We publicly release all our developed software and our complete tool-chain to allow independent experiments and evaluation. This includes our  $\ell$ MBOT decoder



Figure 14:  $\ell$ MBOT rules generating a relative clause/reflexive pronoun

presented in Section 4 and a separate C++ module that we use for rule extraction (see Section 3).

Besides the automatic evaluation, we also performed a small manual analysis of obtained translations and show-cased some examples (see Section 5.3). We argue that our  $\ell$ MBOT approach can adequately handle discontinuous phrases, which occur frequently in German. Other languages that exhibit such phenomena include Czech, Dutch, Russian, and Polish. Thus, we hope that our system can also successfully be applied for other language pairs, which we plan to pursue as well.

In other future work, we want to investigate full backwards application of  $\ell$ MBOT rules, which would be more suitable for the converse translation direction German  $\rightarrow$  English. The current independent LM scoring of components has some negative side-effects that we plan to circumvent with the use of lazy LM scoring.

## Acknowledgement

The authors thank Alexander Fraser for his ongoing support and advice. All authors were financially supported by the German Research Foundation (DFG) grant MA 4959/1-1.

## References

- André Arnold and Max Dauchet. 1982. Morphismes et bimorphismes d'arbres. *Theoret. Comput. Sci.*, 20(1):33–93.
- Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky Wide Web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proc. 4th Workshop on Statistical Machine Translation*, pages 1–28.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proc. 43rd ACL*, pages 173–180.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computat. Linguist.*, 33(2):201–228.
- David Chiang. 2010. Learning to translate with source and target syntax. In *Proc. 48th ACL*, pages 1443–1452.
- Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. 41st ACL*, pages 205–208.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proc. HLT-NAACL*, pages 273–280.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve Deneefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. 44th ACL*, pages 961–968.
- Hieu Hoang, Philipp Koehn, and Adam Lopez. 2009. A unified framework for phrase-based, hierarchical, and syntax-based statistical machine translation. In *Proc. 6th Int. Workshop Spoken Language Translation*, pages 152–159.
- Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. 7th Conf. Association for Machine Translation of the Americas*, pages 66–73.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT-NAACL*, pages 127–133.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT Speech Translation Evaluation. In *Proc. 2nd Int. Workshop Spoken Language Translation*.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. ACL*, pages 177–180.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP*, pages 388–395.
- Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proc. 10th Machine Translation Summit*, pages 79–86.
- Alon Lavie, Alok Parlikar, and Vamshi Ambati. 2008. Syntax-driven learning of sub-sentential translation equivalents and translation rules from parsed parallel corpora. In *Proc. 2nd ACL Workshop on Syntax and Structure in Statistical Translation*, pages 87–95.
- Eric Lilin. 1978. *Une généralisation des transducteurs d'états finis d'arbres: les S-transducteurs*. Thèse 3ème cycle, Université de Lille.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. 44th ACL*, pages 609–616.
- Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proc. 47th ACL*, pages 558–566.
- Andreas Maletti. 2010. Why synchronous tree substitution grammars? In *Proc. HLT-NAACL*, pages 876–884.
- Andreas Maletti. 2011. How to train your multi bottom-up tree transducer. In *Proc. 49th ACL*, pages 825–834.
- Andreas Maletti. 2012. Every sensible extended top-down tree transducer is a multi bottom-up tree transducer. In *Proc. HLT-NAACL*, pages 263–273.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computat. Linguist.*, 29(1):19–51.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. 41st ACL*, pages 160–167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. 40th ACL*, pages 311–318.
- Jean-Claude Raoult. 1997. Rational tree relations. *Bull. Belg. Math. Soc. Simon Stevin*, 4(1):149–176.
- Helmut Schmid. 2004. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proc. 20th COLING*, pages 162–168.

- Jun Sun, Min Zhang, and Chew Lim Tan. 2009. A non-contiguous tree sequence alignment-based model for statistical machine translation. In *Proc. 47th ACL*, pages 914–922.
- Web-as-Corpus Consortium. 2008. SDeWaC — a 0.88 billion word corpus for german. Website: <http://wacky.sslmit.unibo.it/doku.php>.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computat. Linguist.*, 23(3):377–403.
- Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008a. A tree sequence alignment-based tree-to-tree translation model. In *Proc. 46th ACL*, pages 559–567.
- Min Zhang, Hongfei Jiang, Haizhou Li, Aiti Aw, and Sheng Li. 2008b. Grammar comparison study for translational equivalence modeling and statistical machine translation. In *Proc. 22nd International Conference on Computational Linguistics*, pages 1097–1104.

# Enlisting the Ghost: Modeling Empty Categories for Machine Translation

**Bing Xiang**  
IBM T. J. Watson Research Center  
1101 Kitchawan Rd  
Yorktown Heights, NY 10598  
bxiang@us.ibm.com

**Xiaoqiang Luo \***  
Google Inc.  
111 8th Ave  
New York, NY 10011  
xql@google.com

**Bowen Zhou**  
IBM T. J. Watson Research Center  
1101 Kitchawan Rd  
Yorktown Heights, NY 10598  
zhou@us.ibm.com

## Abstract

Empty categories (EC) are artificial elements in Penn Treebanks motivated by the government-binding (GB) theory to explain certain language phenomena such as pro-drop. ECs are ubiquitous in languages like Chinese, but they are tacitly ignored in most machine translation (MT) work because of their elusive nature. In this paper we present a comprehensive treatment of ECs by first recovering them with a structured MaxEnt model with a rich set of syntactic and lexical features, and then incorporating the predicted ECs into a Chinese-to-English machine translation task through multiple approaches, including the extraction of EC-specific sparse features. We show that the recovered empty categories not only improve the word alignment quality, but also lead to significant improvements in a large-scale state-of-the-art syntactic MT system.

## 1 Introduction

One of the key challenges in statistical machine translation (SMT) is to effectively model inherent differences between the source and the target language. Take the Chinese-English SMT as an example: it is non-trivial to produce correct pronouns on the target side when the source-side pronoun is missing. In addition, the pro-drop problem can also degrade the word alignment quality in the training data. A sentence pair observed in the real data is shown in Figure 1 along with the word alignment obtained from an automatic word aligner, where the English subject pronoun

“that” is missing on the Chinese side. Consequently, “that” is incorrectly aligned to the second to the last Chinese word “De”, due to their high co-occurrence frequency in the training data. If the dropped pronoun were recovered, “that” would have been aligned with the dropped-pro (cf. Figure 3), which is a much more sensible alignment.

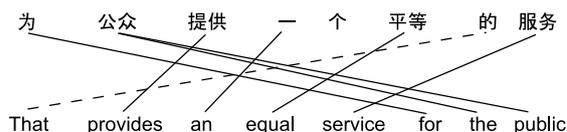


Figure 1: Example of incorrect word alignment due to missing pronouns on the Chinese side.

In order to account for certain language phenomena such as pro-drop and wh-movement, a set of special tokens, called empty categories (EC), are used in Penn Treebanks (Marcus et al., 1993; Bies and Maamouri, 2003; Xue et al., 2005). Since empty categories do not exist in the surface form of a language, they are often deemed elusive and recovering ECs is even figuratively called “chasing the ghost” (Yang and Xue, 2010).

In this work we demonstrate that, with the availability of large-scale EC annotations, it is feasible to predict and recover ECs with high accuracy. More importantly, with various approaches of modeling the recovered ECs in SMT, we are able to achieve significant improvements<sup>1</sup>.

The contributions of this paper include the following:

- Propose a novel structured approach to EC prediction, including the exact word-level lo-

\* This work was done when the author was with IBM.

<sup>1</sup>Hence “Enlisting the ghost” in the title of this paper.

cation and EC labels. Our results are significantly higher in accuracy than that of the state-of-the-art;

- Measure the effect of ECs on automatic word alignment for machine translation after integrating recovered ECs into the MT data;
- Design EC-specific features for phrases and syntactic tree-to-string rules in translation grammar;
- Show significant improvement on top of the state-of-the-art large-scale hierarchical and syntactic machine translation systems.

The rest of the paper is organized as follows. In Section 2, we present a structured approach to EC prediction. In Section 3, we describe the integration of Chinese ECs in MT. The experimental results for both EC prediction and SMT are reported in Section 4. A survey on the related work is conducted in Section 5, and Section 6 summarizes the work and introduces some future work.

## 2 Chinese Empty Category Prediction

The empty categories in the Chinese Treebank (CTB) include trace markers for A'- and A-movement, dropped pronoun, big PRO etc. A complete list of categories used in CTB is shown in Table 1 along with their intended usages. Readers are referred to the documentation (Xue et al., 2005) of CTB for detailed discussions about the characterization of empty categories.

EC	Meaning
*T*	trace of A'-movement
*	trace of A-movement
*PRO*	big PRO in control structures
*pro*	pro-drop
*OP*	operator in relative clauses
*RNR*	for right node raising

Table 1: List of empty categories in the CTB.

In this section, we tackle the problem of recovering Chinese ECs. The problem has been studied before in the literature. For instance, Yang and Xue (2010) attempted to predict the existence of an EC before a word; Luo and Zhao (2011) predicted ECs on parse trees, but the position information of some ECs is partially lost in their representation. Furthermore, Luo and Zhao (2011) conducted experiments on gold parse trees only. In

our opinion, recovering ECs from machine parse trees is more meaningful since that is what one would encounter when developing a downstream application such as machine translation. In this paper, we aim to have a more comprehensive treatment of the problem: all EC types along with their locations are predicted, and we will report the results on both human parse trees and machine-generated parse trees.

### 2.1 Representation of Empty Categories

Our effort of recovering ECs is a two-step process: first, at training time, ECs in the Chinese Treebank are moved and preserved in the portion of the tree structures pertaining to surface words only. Original ECs and their subtrees are then deleted without loss of information; second, a model is trained on transformed trees to predict and recover ECs.

Empty categories heavily depend on syntactic tree structure. For this reason, we choose to project them onto a parse tree node. To facilitate presentation, we first distinguish a *solid* vs. an *empty* non-terminal node. A non-terminal node is *solid* if and only if it contains at least one child node that spans one or more surface words (as opposed to an EC); accordingly, an *empty* node is a non-terminal node that spans only ECs. In the left half of Figure 2, the NP node that is the immediate child of IP has only one child node spanning an EC – (-NONE- \*pro\*), and is thus an *empty* node; while all other non-terminal nodes have at least one surface word as their child and are thus all *solid* nodes.

We decide to attach an EC to its lowest *solid* ancestor node. That is, the EC is moved up to the first solid node in the syntactic tree. After ECs are attached, all empty nodes and ECs are deleted from the tree. In order to uniquely recover ECs, we also need to encode the position information. To this end, the relative child index of an EC is affixed to the EC tag. Take the NP node spanning the \*pro\* in Figure 2 as an example, the \*pro\* is moved to the lowest solid ancestor, IP node, and its position is encoded by @1 since the deleted NP is the second child of the IP node (we use 0-based indices). With this transformation, we are able to recover not only the position of an EC, but its type as well. A special tag NULL is attached to non-terminal nodes without EC. Since an EC is introduced to express the structure of a sentence, it is a good practice to associate it with the syn-

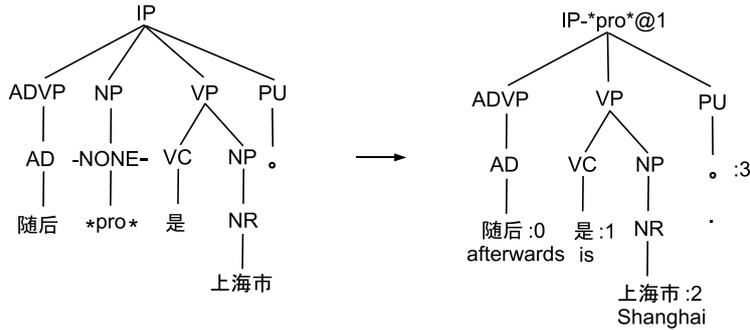


Figure 2: Example of tree transformation on training data to encode an empty category and its position information.

tactic tree, as opposed to simply attaching it to a neighboring word, as was done in (Yang and Xue, 2010). We believe this is one of the reasons why our model has better accuracy than that of (Yang and Xue, 2010) (cf. Table 7).

In summary, a projected tag consists of an EC type (such as `*pro*`) and the EC’s position information. The problem of predicting ECs is then cast into predicting an EC tag at each non-terminal node. Notice that the input to such a predictor is a syntactic tree without ECs, e.g., the parse tree on the right hand of Figure 2 without the EC tag `*pro*@1` is such an example.

## 2.2 A Structured Empty Category Model

We propose a structured MaxEnt model for predicting ECs. Specially, given a syntactic tree,  $T$ , whose ECs have been projected onto solid nodes with the procedure described in Section 2.1, we traverse it in post-order (i.e., child nodes are visited recursively first before the current node is visited). Let  $T = t_1 t_2 \cdots t_n$  be the sequence of nodes produced by the post-order traversal, and  $e_i (i = 1, 2, \cdots, n)$  be the EC tag associated with  $t_i$ . The probabilistic model is then:

$$\begin{aligned}
 P(e_1^n | T) &= \prod_{i=1}^n P(e_i | T, e_1^{i-1}) \\
 &= \prod_{i=1}^n \frac{\exp(\sum_k \lambda_k f_k(e_1^{i-1}, T, e_i))}{Z(e_1^{i-1}, T)} \quad (1)
 \end{aligned}$$

Eq. (1) is the familiar log linear (or MaxEnt) model, where  $f_k(e_1^{i-1}, T, e_i)$  is the feature function and

$Z(e_1^{i-1}, T) = \sum_{e \in \mathcal{E}} \exp(\sum_k \lambda_k f_k(e_1^{i-1}, T, e))$  is the normalization factor.  $\mathcal{E}$  is the set of ECs to be predicted. In the CTB 7.0 processed by the procedure in Section 2.1, the set consists of 32 EC tags

plus a special NULL symbol, obtained by modulating the list of ECs in Table 1 with their positions (e.g., `*pro*@1` in Figure 2).

Once the model is chosen, the next step is to decide a set of features  $\{f_k(e_1^{i-1}, T, e_i)\}$  to be used in the model. One advantage of having the representation in Section 2.1 is that it is very easy to compute features from tree structures. Indeed, all features used in our system are computed from the syntactic trees, including lexical features.

There are 3 categories of features used in the model: (1) tree label features; (2) lexical features; (3) EC features, and we list them in Table 2. In the feature description column, all node positions (e.g., “left”, “right”) are relative to the current node being predicted.

Feature 1 to 10 are computed directly from parse trees, and are straightforward. We include up to 2 siblings when computing feature 9 and 10. Feature 11 to 17 are lexical features. Note that we use words at the edge of the current node: feature 11 and 12 are words at the internal boundary of the current node, while feature 13 and 14 are the immediately neighboring word external to the current node. Feature 15 and 17 are from head word information of the current node and the parent node. Feature 18 and 19 are computed from predicted ECs in the past – that’s why the model in Eq. (1) conditions on  $e_1^{i-1}$ .

Besides the features presented in Table 2, we also use conjunction features between the current node label with the parent node label; the current node label with features computed from child nodes; the current node label with features from left and sibling nodes; the current node label with lexical features.

No.	Tree Label Features
1	current node label
2	parent node label
3	grand-parent node label
4	left-most child label or POS tag
5	right-most child label or POS tag
6	label or POS tag of the head child
7	the number of child nodes
8	one level CFG rule
9	left-sibling label or POS tag
10	right-sibling label or POS tag
<b>Lexical Features</b>	
11	left-most word under the current node
12	right-most word under the current node
13	word immediately left to the span of the current node
14	word immediately right to the span of the current node
15	head word of the current node
16	head word of the parent node
17	is the current node head child of its parent?
<b>EC Features</b>	
18	predicted EC of the left sibling
19	the set of predicted ECs of child nodes

Table 2: List of features.

### 3 Integrating Empty Categories in Machine Translation

In this section, we explore multiple approaches of utilizing recovered ECs in machine translation.

#### 3.1 Explicit Recovery of ECs in MT

We conducted some initial error analysis on our MT system output and found that most of the errors that are related to ECs are due to the missing *\*pro\** and *\*PRO\**. This is also consistent with the findings in (Chung and Gildea, 2010). One of the other frequent ECs, *\*OP\**, appears in the Chinese relative clauses, which usually have a Chinese word “De” aligned to the target side “that” or “which”. And the trace, *\*T\**, exists in both Chinese and English sides. For MT we want to focus on the places where there exist mismatches between the source and target languages. A straightforward way of utilizing the recovered *\*pro\** and *\*PRO\** is to pre-process the MT training and test data by inserting ECs into the original source text (i.e. Chinese in this case). As mentioned in the previous section, the output of our EC predictor is a new parse tree with the labels and positions

encoded in the tags. Based on the positional information in the tags, we can move the predicted ECs down to the surface level and insert them between original source words. The same prediction and “pull-down” procedure can be conducted consistently across the MT training and test data.

#### 3.2 Grammar Extraction on Augmented Data

With the pre-processed MT training corpus, an unsupervised word aligner, such as GIZA++, can be used to generate automatic word alignment, as the first step of a system training pipeline. The effect of inserting ECs is two-fold: first, it can impact the automatic word alignment since now it allows the target-side words, especially the function words, to align to the inserted ECs and fix some errors in the original word alignment; second, new phrases and rules can be extracted from the pre-processed training data. For example, for a hierarchical MT system, some phrase pairs and Hiero (Chiang, 2005) rules can be extracted with recovered *\*pro\** and *\*PRO\** at the Chinese side.

In this work we also take advantages of the augmented Chinese parse trees (with ECs projected to the surface) and extract tree-to-string grammar (Liu et al., 2006) for a tree-to-string MT system. Due to the recovered ECs in the source parse trees, the tree-to-string grammar extracted from such trees can be more discriminative, with an increased capability of distinguishing different context. An example of an augmented Chinese parse tree aligned to an English string is shown in Figure 3, in which the incorrect alignment in Figure 1 is fixed. A few examples of the extracted Hiero rules and tree-to-string rules are also listed, which we would not have been able to extract from the original incorrect word alignment when the *\*pro\** was missing.

#### 3.3 Soft Recovery: EC-Specific Sparse Features

Recovered ECs are often good indicators of what hypothesis should be chosen during decoding. In addition to the augmented syntax-based grammar, we propose sparse features as a soft constraint to boost the performance. For each phrase pair, Hiero rule or tree-to-string rule in the MT system, a binary feature  $f_k$  fires if there exists a *\*pro\** on the source side and it aligns to one of its most frequently aligned target words found in the training corpus. We also fire another feature if *\*pro\**

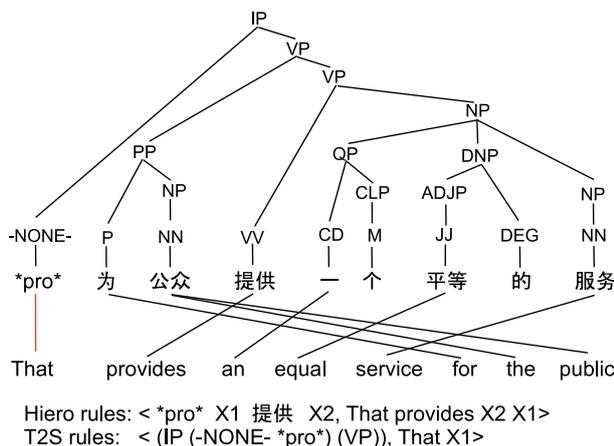


Figure 3: Fixed word alignment and examples of extracted Hiero rules and tree-to-string rules.

aligns to any other target words so the model can choose to penalize them based on a tuning set. Similar features can fire for \*PRO\*. The feature weights can be tuned on a tuning set in a log-linear model along with other usual features/costs, including language model scores, bi-direction translation probabilities, etc. The motivation for such sparse features is to reward those phrase pairs and rules that have highly confident lexical pairs specifically related to ECs, and penalize those who don't have such lexical pairs.

Table 3 listed some of the most frequent English words aligned to \*pro\* or \*PRO\* in a Chinese-English parallel corpus with 2M sentence pairs. Their co-occurrence counts and the lexical translation probabilities are also shown in the table. In total we use 15 sparse features for frequent lexical pairs, including 13 for \*pro\* and 2 for \*PRO\*, and two more features for any other target words that align to \*pro\* or \*PRO\*.

Source	Target	Counts	$P(t s)$
*pro*	the	93100	0.11
*pro*	to	86965	0.10
*pro*	it	45423	0.05
*pro*	in	36129	0.04
*pro*	we	24509	0.03
*pro*	which	17259	0.02
*PRO*	to	195464	0.32
*PRO*	for	31200	0.05

Table 3: Example of frequent word pairs used for sparse features.

## 4 Experimental Results

### 4.1 Empty Category Prediction

We use Chinese Treebank (CTB) v7.0 to train and test the EC prediction model. We partition the data into training, development and test sets. The training set includes 32925 sentences from CTB files 0001-0325, 0400-0454, 0500-0542, 0600-0840, 0590-0596, 1001-1120, 2000-3000, cctv, cnn, msnbc, and phoenix 00-06. The development set has 3033 sentences, from files 0549-0554, 0900-0931, 1136-1151, 3076-3145, and phoenix 10-11. The test set contains 3297 sentences, from files 0543-0548, 0841-0885, 1121-1135, 3001-3075, and phoenix 07-09.

To measure the accuracy of EC prediction, we project the predicted tags from the upper level nodes in the parse trees down to the surface level based on the position information encoded in the tags. The position index for each inserted EC, counted at the surface level, is attached for scoring purpose. The same operation is applied on both the reference and the system output trees. Such projection is necessary, especially when the two trees differ in structure (e.g. gold trees vs. machine-generated trees). We compute the precision, recall and F1 scores for each EC on the test set, and collect their counts in the reference and system output. The results are shown in Table 4, where the LDC gold parse trees are used to extract syntactic features for the model. The first row in the table shows the accuracy for the places where no EC should be inserted. The predictor achieves 99.5% F1 score for this category, with limited number of missing or false positives. The F1 scores for majority of the ECs are above 70%, except for “\*”, which is relatively rare in the data. For the two categories that are interesting to MT, \*pro\* and \*PRO\*, the predictor achieves 74.3% and 81.5% in F1 scores, respectively.

The results reported above are based on the LDC gold parse trees. To apply the EC prediction to NLP applications, such as MT, it is impossible to always rely on the gold trees due to its limited availability. We parse our test set with a maximum entropy based statistical parser (Ratnaparkhi, 1997) first. The parser accuracy is around 84% on the test set. Then we extract features based on the system-generated parse trees, and decode with the previously trained model. The results are shown in Table 5. Compared to those in Table 4, the F1 scores dropped by different degrees for dif-

Tag	Ref	Sys	P	R	F1
NULL	75159	75508	99.3	99.7	99.5
*pro*	1692	1442	80.8	68.9	74.3
*PRO*	1410	1282	85.6	77.8	81.5
*T*	1851	1845	82.8	82.5	82.7
*OP*	1721	1853	90.9	97.9	94.2
*RNR*	51	39	87.2	66.7	75.6
*	156	96	63.5	39.1	48.4

Table 4: Prediction accuracy with gold parse trees, where NULL represents the cases where no ECs should be produced.

ferent types. Such performance drop is expected since the system relies heavily on syntactic structure, and parsing errors create an inherent mismatching condition between the training and testing time. The smallest drop among all types is on NULL, at about 1.6%. The largest drop occurs for \*OP\*, at 27.1%, largely due to the parsing errors on the CP nodes. The F1 scores for \*pro\* and \*PRO\* when using system-generated parse trees are between 50% to 60%.

Tag	Precision	Recall	F1
NULL	97.6	98.2	97.9
*pro*	51.1	50.1	50.6
*PRO*	66.4	50.5	57.3
*T*	68.2	59.9	63.8
*OP*	66.8	67.3	67.1
*RNR*	70.0	54.9	61.5
*	60.9	35.9	45.2

Table 5: Prediction accuracy with system-generated parse trees.

To show the effect of ECs other than \*pro\* and \*PRO\*, we remove all ECs in the training data except \*pro\* and \*PRO\*. So the model only predicts NULL, \*pro\* or \*PRO\*. The results on the test set are listed in Table 6. There is 0.8% and 0.5% increase on NULL and \*pro\*, respectively. The F1 score for \*PRO\* drops by 0.2% slightly.

As mentioned earlier, for MT we focus on recovering \*pro\* and \*PRO\* only. The model generating the results in Table 6 is the one we applied in our MT experiments reported later.

In order to compare to the state-of-the-art models to see where our model stands, we switch our training, development and test data to those used in the work of (Yang and Xue, 2010) and (Cai et

Tag	Precision	Recall	F1
NULL	98.5	98.9	98.7
*pro*	51.0	51.1	51.1
*PRO*	66.0	50.4	57.1

Table 6: Prediction accuracy with system-generated parse trees, modeling \*pro\* and \*PRO\* only.

al., 2011), for the purpose of a direct comparison. The training set includes CTB files 0081 through 0900. The development set includes files 0041 to 0080, and the test set contains files 0001-0040 and 0901-0931. We merge all empty categories into a single type in the training data before training our EC prediction model. To compare the performance on system-generated parse trees, we also train a Berkeley parser on the same training data and parse the test set. The prediction accuracy for such single type on the test set with gold or system-generated parse trees is shown in Table 7, compared to the numbers reported in (Yang and Xue, 2010) and (Cai et al., 2011). The model we proposed achieves 6% higher F1 score than that in (Yang and Xue, 2010) and 2.6% higher than that in (Cai et al., 2011), which is significant. This shows the effectiveness of our structured approach.

Model	T	P	R	F1
(Yang and Xue, 2010)	G	95.9	83.0	89.0
Structured (this work)	G	96.5	93.6	95.0
(Yang and Xue, 2010)	S	80.3	52.1	63.2
(Cai et al., 2011)	S	74.0	61.3	67.0
Structured (this work)	S	74.9	65.1	69.6

Table 7: Comparison with the previous results, using the same training and test data. T: parse trees. G: gold parse trees. S: system-generated parse trees. P: precision. R: recall.

## 4.2 MT Results

In the Chinese-to-English MT experiments, we test two state-of-the-art MT systems. One is an re-implementation of Hiero (Chiang, 2005), and the other is a hybrid syntax-based tree-to-string system (Zhao and Al-onaizan, 2008), where normal phrase pairs and Hiero rules are used as a backoff for tree-to-string rules.

The MT training data includes 2 million sentence pairs from the parallel corpora released by

LDC over the years, with the data from United Nations and Hong Kong excluded <sup>2</sup>. The Chinese text is segmented with a segmenter trained on the CTB data using conditional random field (CRF), followed by the longest-substring match segmentation in a second pass. Our language model (LM) training data consists of about 10 billion English words, which includes Gigaword and other newswire and web data released by LDC, as well as the English side of the parallel training corpus. We train a 6-gram LM with modified Kneser-Ney smoothing (Chen and Goodman, 1998). Our tuning set for MT contains 1275 sentences from LDC2010E30. We test our system on the NIST MT08 Newswire (691 sentences) and Weblog (666 sentences) sets. Both tuning and test sets have 4 sets of references for each sentence. The MT systems are optimized with pairwise ranking optimization (Hopkins and May, 2011) to maximize BLEU (Papineni et al., 2002).

We first predict \*pro\* and \*PRO\* with our annotation model for all Chinese sentences in the parallel training data, with \*pro\* and \*PRO\* inserted between the original Chinese words. Then we run GIZA++ (Och and Ney, 2000) to generate the word alignment for each direction and apply grow-diagonal-final (Koehn et al., 2003), same as in the baseline. We want to measure the impact on the word alignment, which is an important step for the system building. We append a 300-sentence set, which we have human hand alignment available as reference, to the 2M training sentence pairs before running GIZA++. The alignment accuracy measured on this alignment test set, with or without \*pro\* and \*PRO\* inserted before running GIZA++, is shown in Table 8. To make a fair comparison with the baseline alignment, any target words aligned to ECs are deemed as unaligned during scoring. We observe 1.2% improvement on function word related links, and almost the same accuracy on content words. This is understandable since \*pro\* and \*PRO\* are mostly aligned to the function words at the target side. The precision and recall for function words are shown in Table 9. We can see higher accuracy in both precision and recall when ECs (\*pro\* and \*PRO\*) are recovered in the Chinese side. Especially, the precision is improved by 2% absolute.

<sup>2</sup>The training corpora include LDC2003E07, LDC2003E08, LDC2005T10, LDC2006E26, LDC2006G05, LDC2007E103, LDC2008G05, LDC2009G01, and LDC2009G02.

System	Function	Content	All
Baseline	51.7	69.7	65.4
+EC	52.9	69.6	65.7

Table 8: Word alignment F1 scores with or without \*pro\* and \*PRO\*.

System	Precision	Recall	F1
Baseline	54.1	49.5	51.7
+EC	56.0	50.1	52.9

Table 9: Word alignment accuracy for function words only.

Next we extract phrase pairs, Hiero rules and tree-to-string rules from the original word alignment and the improved word alignment, and tune all the feature weights on the tuning set. The weights include those for usual costs and also the sparse features proposed in this work specifically for ECs. We test all the systems on the MT08 Newswire and Weblog sets.

The BLEU scores from different systems are shown in Table 10 and Table 11, respectively. We measure the incremental effect of prediction (inserting \*pro\* and \*PRO\*) and sparse features. Pre-processing of the data with ECs inserted improves the BLEU scores by about 0.6 for newswire and 0.2 to 0.3 for the weblog data, compared to each baseline separately. On top of that, adding sparse features helps by another 0.3 on newswire and 0.2 to 0.4 on weblog. Overall, the Hiero and tree-to-string systems are improved by about 1 point for newswire and 0.4 to 0.7 for weblog. The smaller gain on the weblog data could be due to the more difficult data to parse, which affects the accuracy of EC prediction. All the results in Table 10 and 11 marked with “\*” are statistically significant with  $p < 0.05$  using the sign test described in (Collins et al., 2005), compared to the baseline results in each table. Two MT examples are given in Table 12, which show the effectiveness of the recovered ECs in MT.

System	MT08-nw	MT08-wb
Hiero	33.99	25.40
+prediction	34.62*	25.63
+prediction+sparse	34.95*	25.80*

Table 10: BLEU scores in the Hiero system.

Source	*pro* 要记住 , " 我们 是 幸运 的 一 代 " 。
Reference	You should remember, "We are the fortunate generation."
Old	Have to remember that "We are the fortunate generation."
New	You have to remember that "We are the fortunate generation."
Source	塔利班 呼吁 德国 与 韩国 *PRO* 自 阿富汗 撤军 ,
Reference	The Taliban called on Germany and South Korea to withdraw their troops from Afghanistan,
Old	The Taliban called on Germany and South Korea withdraw their troops from Afghanistan,
New	The Taliban called on Germany and South Korea to withdraw their troops from Afghanistan,

Table 12: Examples of baseline (old) and improved system (new) output.

System	MT08-nw	MT08-wb
T2S+Hiero	34.53	25.80
+prediction	35.17*	26.08
+prediction+sparse	35.51*	26.53*

Table 11: BLEU scores in the tree-to-string system with Hiero rules as backoff.

## 5 Related Work

Empty categories have been studied in recent years for several languages, mostly in the context of reference resolution and syntactic processing for English, such as in (Johnson, 2002; Dienes and Dubey, 2003; Gabbard et al., 2006). More recently, EC recovery for Chinese started emerging in literature. In (Guo et al., 2007), non-local dependencies are migrated from English to Chinese for generating proper predicate-argument-modifier structures from surface context free phrase structure trees. In (Zhao and Ng, 2007), a decision tree learning algorithm is presented to identify and resolve Chinese anaphoric zero pronouns. and achieves a performance comparable to a heuristic rule-based approach. Similar to the work in (Dienes and Dubey, 2003), empty detection is formulated as a tagging problem in (Yang and Xue, 2010), where each word in the sentence receives a tag indicating whether there is an EC before it. A maximum entropy model is utilized to predict the tags, but different types of ECs are not distinguished. In (Cai et al., 2011), a language-independent method was proposed to integrate the recovery of empty elements into syntactic parsing. As shown in the previous section, our model outperforms the model in (Yang and Xue, 2010) and (Cai et al., 2011) significantly using the same training and test data. (Luo and Zhao, 2011) also tries to predict the existence of an EC

in Chinese sentences, but the ECs in the middle of a tree constituent are lumped into a single position and are not uniquely recoverable.

There exists only a handful of previous work on applying ECs explicitly to machine translation so far. One of them is the work reported in (Chung and Gildea, 2010), where three approaches are compared, based on either pattern matching, CRF, or parsing. However, there is no comparison between using gold trees and automatic trees. There also exist a few major differences on the MT part between our work and theirs. First, in addition to the pre-processing of training data and inserting recovered empty categories, we implement sparse features to further boost the performance, and tune the feature weights directly towards maximizing the machine translation metric. Second, there is no discussion on the quality of word alignment in (Chung and Gildea, 2010), while we show the alignment improvement on a hand-aligned set. Last, they use a phase-based system trained on only 60K sentences, while we conduct experiments on more advanced Hiero and tree-to-string systems, trained on 2M sentences in a much larger corpus. We directly take advantage of the augmented parse trees in the tree-to-string grammar, which could have larger impact on the MT system performance.

## 6 Conclusions and Future Work

In this paper, we presented a novel structured approach to EC prediction, which utilizes a maximum entropy model with various syntactic features and shows significantly higher accuracy than the state-of-the-art approaches. We also applied the predicted ECs to a large-scale Chinese-to-English machine translation task and achieved significant improvement over two strong MT base-

lines, i.e. a hierarchical phase-based system and a tree-to-string syntax-based system. More work remain to be done next to further take advantages of ECs. For example, the recovered ECs can be encoded in a forest as the input to the MT decoder and allow the decoder to pick the best MT output based on various features in addition to the sparse features we proposed in this work. Many promising approaches can be explored in the future.

## Acknowledgments

We would like to acknowledge the support of DARPA under Grant HR0011-12-C-0015 for funding part of this work. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

## References

- Ann Bies and Mohamed Maamouri. 2003. Penn Arabic treebank guidelines. In <http://www.ircs.upenn.edu/arabic/Jan03release/guidelines-TB-1-28-03.pdf>.
- Shu Cai, David Chiang, and Yoav Goldberg. 2011. Language-independent parsing with empty elements. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: short papers*, pages 212–216.
- Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. In *Technical Report TR-10-98, Computer Science Group, Harvard University*.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Ann Arbor, Michigan, June.
- Tagyoung Chung and Daniel Gildea. 2010. Effects of empty categories on machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 636–645.
- Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531–540.
- Peter Dienes and Amit Dubey. 2003. Deep syntactic processing by combining shallow methods. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.
- Ryan Gabbard, Seth Kulick, and Mitchell Marcus. 2006. Fully parsing the penn treebank. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.
- Yuqing Guo, Haifeng Wang, and Josef van Genabith. 2007. Recovering non-local dependencies for Chinese. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362.
- Mark Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.
- Philipp Koehn, Franz Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL*, pages 48–54.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616.
- Xiaoqiang Luo and Bing Zhao. 2011. A statistical tree annotator and its applications. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1230–1238.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. In *Computational Linguistics*, volume 19(2), pages 313–330.
- Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, China, October.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.
- Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of Second Conference on Empirical*

*Methods in Natural Language Processing*, pages 1–10.

Nianwen Xue, Fei Xia, Fu dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*, volume 11(2), pages 207–238.

Yaqin Yang and Nianwen Xue. 2010. Chasing the ghost: Recovering empty categories in the Chinese Treebank. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1382–1390, Beijing, China, August.

Bing Zhao and Yaser Al-onaihan. 2008. Generalizing local and non-local word-reordering patterns for syntax-based machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 572–581.

Shanheng Zhao and Hwee Tou Ng. 2007. Identification and resolution of Chinese zero pronouns: A machine learning approach. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

# A Multi-Domain Translation Model Framework for Statistical Machine Translation

**Rico Sennrich**

Institute of Computational Linguistics  
University of Zurich  
Binzmühlestr. 14  
CH-8050 Zürich  
sennrich@cl.uzh.ch

**Holger Schwenk and Walid Aransa**

LIUM, University of Le Mans  
72085 Le Mans cedex 9, France  
lastname@lium.univ-lemans.fr

## Abstract

While domain adaptation techniques for SMT have proven to be effective at improving translation quality, their practicality for a multi-domain environment is often limited because of the computational and human costs of developing and maintaining multiple systems adapted to different domains. We present an architecture that delays the computation of translation model features until decoding, allowing for the application of mixture-modeling techniques at decoding time. We also describe a method for unsupervised adaptation with development and test data from multiple domains. Experimental results on two language pairs demonstrate the effectiveness of both our translation model architecture and automatic clustering, with gains of up to 1 BLEU over unadapted systems and single-domain adaptation.

## 1 Introduction

The effectiveness of domain adaptation approaches such as mixture-modeling (Foster and Kuhn, 2007) has been established, and has led to research on a wide array of adaptation techniques in SMT, for instance (Matsoukas et al., 2009; Shah et al., 2012). In all these approaches, adaptation is performed during model training, with respect to a representative development corpus, and the models are kept unchanged when the system is deployed. Therefore, when working with multiple and/or unlabelled domains, domain adaptation is often impractical for a number of reasons. Firstly, maintaining multiple systems for each language pair, each adapted to a different domain, is costly

in terms of computational and human resources: the full system development pipeline needs to be performed for all identified domains, all the models are separately stored and need to be switched at runtime. This is impractical in many real applications, in particular a web translation service which is faced with texts coming from many different domains. Secondly, domain adaptation bears a risk of performance loss. If there is a mismatch between the domain of the development set and the test set, domain adaptation can potentially harm performance compared to an unadapted baseline.

We introduce a translation model architecture that delays the computation of features to the decoding phase. The calculation is based on a vector of component models, with each component providing the sufficient statistics necessary for the computation of the features. With this framework, adaptation to a new domain simply consists of updating a weight vector, and multiple domains can be supported by the same system.

We also present a clustering approach for unsupervised adaptation in a multi-domain environment. In the development phase, a set of development data is clustered, and the models are adapted to each cluster. For each sentence that is being decoded, we choose the weight vector that is optimized on the closest cluster, allowing for adaptation even with unlabelled and heterogeneous test data.

## 2 Related Work

(Ortiz-Martínez et al., 2010) delay the computation of translation model features for the purpose of interactive machine translation with online training. The main difference to our approach is that we store sufficient statistics not for a single model, but a vector of models, which allows us to

weight the contribution of each component model to the feature calculation. The similarity suggests that our framework could also be used for interactive learning, with the ability to learn a model incrementally from user feedback, and weight it differently than the static models, opening new research opportunities.

(Sennrich, 2012b) perform instance weighting of translation models, based on the sufficient statistics. Our framework implements this idea, with the main difference that the actual combination is delayed until decoding, to support adaptation to multiple domains in a single system.

(Razmara et al., 2012) describe an ensemble decoding framework which combines several translation models in the decoding step. Our work is similar to theirs in that the combination is done at runtime, but we also delay the computation of translation model probabilities, and thus have access to richer sufficient statistics. In principle, our architecture can support all mixture operations that (Razmara et al., 2012) describe, plus additional ones such as forms of instance weighting, which are not possible after the translation probabilities have been computed.

(Banerjee et al., 2010) focus on the problem of domain identification in a multi-domain setting. They use separate translation systems for each domain, and a supervised setting, whereas we aim for a system that integrates support for multiple domains, with or without supervision.

(Yamamoto and Sumita, 2007) propose unsupervised clustering at both training and decoding time. The training text is divided into a number of clusters, a model is trained on each, and during decoding, each sentence is assigned to the closest cluster-specific model. Our approach bears resemblance to this clustering, but is different in that Yamamoto and Sumita assign each sentence to the closest model, and use this model for decoding, whereas in our approach, each cluster is associated with a mixture of models that is optimized to the cluster, and the number of clusters need not be equal to the number of component models.

### 3 Translation Model Architecture

This section covers the architecture of the multi-domain translation model framework. Our translation model is embedded in a log-linear model as is common for SMT, and treated as a single translation model in this log-linear combination. We im-

plemented this architecture for phrase-based models, and will use this terminology to describe it, but in principle, it can be extended to hierarchical or syntactic models.

The architecture has two goals: move the calculation of translation model features to the decoding phase, and allow for multiple knowledge sources (e.g. bitexts or user-provided data) to contribute to their calculation. Our immediate purpose for this paper is domain adaptation in a multi-domain environment, but the delay of the feature computation has other potential applications, e.g. in interactive MT.

We are concerned with calculating four features during decoding, henceforth just referred to as the translation model features:  $p(\bar{s}|\bar{t})$ ,  $lex(\bar{s}|\bar{t})$ ,  $p(\bar{t}|\bar{s})$  and  $lex(\bar{t}|\bar{s})$ .  $\bar{s}$  and  $\bar{t}$  denote the source and target phrase. We follow the definitions in (Koehn et al., 2003).

Traditionally, the phrase translation probabilities  $p(\bar{s}|\bar{t})$  and  $p(\bar{t}|\bar{s})$  are estimated through unsmoothed maximum likelihood estimation (MLE).

$$p(x|y) = \frac{c(x, y)}{c(y)} = \frac{c(x, y)}{\sum_{x'} c(x', y)} \quad (1)$$

where  $c$  denotes the count of an observation, and  $p$  the model probability.

The lexical weights  $lex(\bar{s}|\bar{t})$  and  $lex(\bar{t}|\bar{s})$  are calculated as follows, using a set of word alignments  $a$  between  $\bar{s}$  and  $\bar{t}$ :<sup>1</sup>

$$lex(\bar{s}|\bar{t}, a) = \prod_{i=1}^n \frac{1}{|\{j | (i, j) \in a\}|} \sum_{\forall (i, j) \in a} w(s_i | t_j) \quad (2)$$

A special NULL token is added to  $\bar{t}$  and aligned to each unaligned word in  $\bar{s}$ .  $w(s_i | t_j)$  is calculated through MLE, as in equation 1, but based on the word (pair) frequencies.

To combine statistics from a vector of  $n$  component corpora, we can use a weighted version of equation 1, which adds a weight vector  $\lambda$  of length  $n$  (Sennrich, 2012b):

$$p(x|y; \lambda) = \frac{\sum_{i=1}^n \lambda_i c_i(x, y)}{\sum_{i=1}^n \sum_{x'} \lambda_i c_i(x', y)} \quad (3)$$

The word translation probabilities  $w(t_i | s_j)$  are defined analogously, and used in equation 2 for a weighted version.

<sup>1</sup>The equation shows  $lex(\bar{s}|\bar{t})$ ;  $lex(\bar{t}|\bar{s})$  is computed analogously.

In order to compute the translation model features online, a number of sufficient statistics need to be accessible at decoding time. For  $p(\bar{s}|\bar{t})$  and  $p(\bar{t}|\bar{s})$ , we require the statistics  $c(\bar{s})$ ,  $c(\bar{t})$  and  $c(\bar{s}, \bar{t})$ . For accessing them during decoding, we simply store them in the decoder’s data structure, rather than storing pre-computed translation model features. This means that we can use existing, compact data formats for storing and accessing them.<sup>2</sup>

The statistics are accessed when the decoder collects all translation options for a phrase  $\bar{s}$  in the source sentence. We then access all translation options for each component table, obtaining a vector of statistics  $c(\bar{s})$  for the source phrase, and  $c(\bar{t})$  and  $c(\bar{s}, \bar{t})$  for each potential target phrase. For phrase pairs which are not found,  $c(\bar{s}, \bar{t})$  and  $c(\bar{t})$  are initially set to 0.

Note that  $c(\bar{t})$  is potentially incorrect at this point, since a phrase pair not being found does not entail that  $c(\bar{t})$  is 0. After all tables have been accessed, and we thus know the full set of possible translation options  $(\bar{s}, \bar{t})$ , we perform a second round of lookups for all  $c(\bar{t})$  in the vector which are still set to 0. We introduce a second table for accessing  $c(\bar{t})$  efficiently, again storing it in the decoder’s data structure. We can easily create such a table by inverting the source and target phrases, deduplicating it for compactness (we only need one entry per target phrase), and storing  $c(\bar{t})$  as only feature.

For  $lex(\bar{s}|\bar{t})$ , we require an alignment  $a$ , plus  $c(t_j)$  and  $c(s_i, t_j)$  for all pairs  $(i, j)$  in  $a$ .  $lex(\bar{t}|\bar{s})$  can be based on the same alignment  $a$  (with the exception of NULL alignments, which can be added online), but uses statistics  $c(s_j)$  and  $c(t_i, s_j)$ . For estimating the lexical probabilities, we load the frequencies into a vector of four hash tables.<sup>3</sup>

Both space and time complexity of the lookup is linear to the number of component tables. We deem it is still practical because the collection of translation options is typically only a small fraction of total decoding time, with search making up the largest part. For storing and accessing the sufficient statistics (except for the word (pair) frequencies), we use an on-disk data structure pro-

<sup>2</sup>We have released an implementation of the architecture as part of the Moses decoder.

<sup>3</sup> $c(s, t)$  and  $c(t, s)$  are not identical since the lexical probabilities are based on the unsymmetrized word alignment frequencies (in the Moses implementation which we implement).

phrase (pair)	$c_1(x)$	$c_2(x)$
row	300	80
(row, Zeile)	240	20
(row, Reihe)	60	60
$\lambda$	$p(\text{Zeile} \text{row})$	$p(\text{Reihe} \text{row})$
(1, 1)	0.68	0.32
(1, 10)	0.40	0.60
(10, 1)	0.79	0.21

Table 1: Illustration of instance weighting with weight vectors for two corpora.

vided by Moses, which reduces the memory requirements. Still, the number of components may need to be reduced, for instance through clustering of training data (Sennrich, 2012a).

With a small modification, our framework could be changed to use a single table that stores a vector of  $n$  statistics instead of a vector of  $n$  tables. While this would be more compact in terms of memory, and keep the number of table lookups independent of the number of components, we chose a vector of  $n$  tables for its flexibility. With a vector of tables, tables can be quickly added to or removed from the system (conceivable even at runtime), and can be polymorph. One applications where this could be desirable is interactive machine translation, where one could work with a mix of compact, static tables, and tables designed to be incrementally trainable.

In the unweighted variant, the resulting features are equivalent to training on the concatenation of all training data, excepting differences in word alignment, pruning<sup>4</sup> and rounding. The architecture can thus be used as a drop-in replacement for a baseline system that is trained on concatenated training data, with non-uniform weights only being used for texts for which better weights have been established. This can be done either using domain labels or unsupervised methods as described in the next section.

As a weighted combination method, we implemented instance weighting as described in equation 3. Table 1 shows the effect of weighting two corpora on the probability estimates for the translation of *row*. German *Zeile* (row in a table) is predominant in a bitext from the domain IT, whereas

<sup>4</sup>We prune the tables to the most frequent 50 phrase pairs per source phrase before combining them, since calculating the features for all phrase pairs of very common source phrases causes a significant slow-down. We found that this had no significant effects on BLEU.

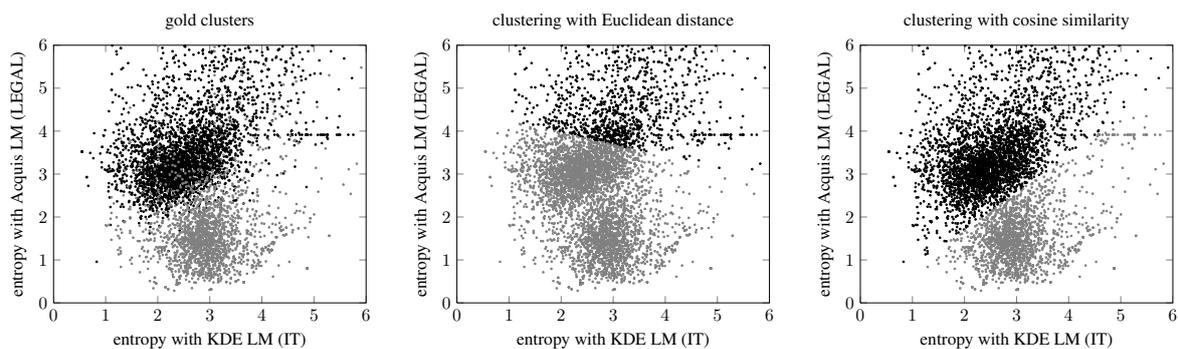


Figure 1: Clustering of data set which contains sentences from two domains: LEGAL and IT. Comparison between gold segmentation, and clustering with two alternative distance/similarity measures. Black: IT; grey: LEGAL.

*Reihe* (line of objects) occurs more often in a legal corpus. Note that the larger corpus (or more precisely, the one in which *row* occurs more often) has a stronger impact on the probability distribution with uniform weights (or in a concatenation of data sets). Instance weighting allows us to modify the contribution of each corpus. In our implementation, the weight vector is set globally, but can be overridden on a per-sentence basis. In principle, using different weight vectors for different phrase pairs in a sentence is conceivable. The framework can also be extended to support other combination methods, such as a linear interpolation of models.

#### 4 Unsupervised Clustering for Online Translation Model Adaptation

The framework supports decoding each sentence with a separate weight vector of size  $4n$ , 4 being the number of translation model features whose computation can be weighted, and  $n$  the number of model components. We now address the question of how to automatically select good weights in a multi-domain task. As a way of optimizing instance weights, (Sennrich, 2012b) minimize translation model perplexity on a set of phrase pairs, automatically extracted from a parallel development set. We follow this technique, but want to have multiple weight vectors, adapted to different texts, between which the system switches at decoding time. The goal is to perform domain adaptation without requiring domain labels or user input, neither for development nor decoding.

The basic idea consists of three steps:

1. Cluster a development set into  $k$  clusters.
2. Optimize translation model weights for each

cluster.

3. For each sentence in the test set, assign it to the nearest cluster and use the translation model weights associated with the cluster.

For step 2, we use the algorithm by (Sennrich, 2012b), implemented in the decoder to allow for a quick optimization of a running system. We will here discuss steps 1 and 3 in more detail.

##### 4.1 Clustering the Development Set

We use  $k$ -means clustering to cluster the sentences of the development set. We train a language model on the source language side of each of the  $n$  component bitexts, and compute an  $n$ -dimensional vector for each sentence by computing its entropy with each language model. Our aim is not to discriminate between sentences that are more likely and unlikely in general, but to cluster on the basis of relative differences between the language model entropies. For this purpose, we choose the cosine as our similarity measure. Figure 1 illustrates clustering in a two-dimensional vector space, and demonstrates that Euclidean distance is unsuitable because it may perform a clustering that is irrelevant to our purposes.

As a result of development set clustering, we obtain a bitext for each cluster, which we use to optimize the model weights, and a centroid per cluster. At decoding time, we need only perform an assignment step. Each test set sentence is assigned to the centroid that is closest to it in the vector space.

##### 4.2 Scalability Considerations

Our theoretical expectation is that domain adaptation will fail to perform well if the test data is from

a different domain than the development data, or if the development data is a heterogeneous mix of domains. A multi-domain setup can mitigate this risk, but only if the relevant domain is represented in the development data, and if the development data is adequately segmented for the optimization. We thus suggest that the development data should contain enough data from all domains that one wants to adapt to, and a high number of clusters.

While the resource requirements increase with the number of component models, increasing the number of clusters is computationally cheap at runtime. Only the clustering of the development set and optimization of the translation model weights for each clusters is affected by  $k$ . This means that the approach can in principle be scaled to a high number of clusters, and support a high number of domains.<sup>5</sup>

The biggest risk of increasing the number of clusters is that if the clusters become too small, perplexity minimization may overfit these small clusters. We will experiment with different numbers of clusters, but since we expect the optimal number of clusters to depend on the amount of development data, and the number of domains, we cannot make generalized statements about the ideal number of  $k$ .

While it is not the focus of this paper, we also evaluate language model adaptation. We perform a linear interpolation of models for each cluster, with interpolation coefficients optimized using perplexity minimization on the development set. The cost of moving language model interpolation into the decoding phase is far greater than for translation models, since the number of hypotheses that need to be evaluated by the language model is several orders of magnitudes higher than the number of phrase pairs used during the translation. For the experiments with language model adaptation, we have chosen to perform linear interpolation offline, and perform language model switching during decoding. While model switching is a fast operation, it also makes the space complexity of storing the language models linear to the number of clusters. For scaling the approach to a high number of clusters, we envision that multi-

<sup>5</sup>If the development set is labelled, one can also use a gold segmentation of development sets instead of  $k$ -means clustering. At decoding time, cluster assignment can be performed by automatically assigning each sentence to the closest centroid, or again through gold labels, if available.

data set	sentences	words (de)
kde	216 000	1 990 000
kdedoc	2880	41 000
kdegb	51 300	450 000
oo	41 000	434 000
oo3	56 800	432 000
php	38 500	301 000
tm	146 000	2 740 000
acquis	2 660 000	58 900 000
dgt	372 000	8 770 000
ecb	110 000	2 850 000
ep7	1 920 000	50 500 000
nc7	159 000	3 950 000
total (train)	5 780 000	131 000 000
dev (IT)	3500	47 000
dev (LEGAL)	2000	46 800
test (IT)	5520	51 800
test (LEGAL)	9780	250 000

Table 2: Parallel data sets English–German.

data set	sentences	words (en)
eu	1 270 000	25 600 000
fiction	830 000	13 700 000
navajo	30 000	490 000
news	110 000	2 550 000
paraweb	370 000	3 930 000
subtitles	2 840 000	21 200 000
techdoc	970 000	7 270 000
total (train)	6 420 000	74 700 000
dev	3500	50 700
test	3500	49 600

Table 3: Parallel data sets Czech–English.

pass decoding, with an unadapted language model in the first phase, and rescoring with a language model adapted online, could perform adequately, and keep the complexity independent of the number of clusters.

## 5 Evaluation

### 5.1 Data and Methods

We conduct all experiments with Moses (Koehn et al., 2007), SRILM (Stolcke, 2002), and GIZA++ (Och and Ney, 2003). Log-linear weights are optimized using MERT (Och and Ney, 2003). We keep the word alignment and lexical reordering models constant through the experiments to minimize the number of confounding factors. We report translation quality using BLEU (Papineni et

system	TM adaptation		LM adaptation		TM+LM adaptation	
	IT	LEGAL	IT	LEGAL	IT	LEGAL
baseline	21.1	49.9	21.1	49.9	21.1	49.9
1 cluster (no split)	21.3*	49.9	21.8*	49.7	21.8*	49.8
2 clusters	21.6*	49.9	22.2*	50.4*	22.8*	50.2*
4 clusters	21.7*	49.9	23.1*	50.2*	22.6*	50.2*
8 clusters	22.1*	49.9	23.1*	50.1*	22.7*	50.3*
16 clusters	21.1	49.9	22.6*	50.3*	21.9*	50.1*
gold clusters	21.8*	50.1*	22.4*	50.1*	23.2*	49.9

Table 4: Translation experiments EN–DE. BLEU scores reported.

al., 2002). We account for optimizer instability by running 3 independent MERT runs per system, and performing significance testing with MultEval (Clark et al., 2011). Systems significantly better than the baseline with  $p < 0.01$  are marked with (\*).

We conduct experiments on two data sets. The first is an English–German translation task with two domains, texts related to information technology (IT) and legal documents (LEGAL). We use data sets from both domains, plus out-of-domain corpora, as shown in table 2. 7 data sets come from the domain IT: 6 from OPUS (Tiedemann, 2009) and a translation memory (tm) provided by our industry partner. 3 data sets are from the legal domain: the ECB corpus from OPUS, plus the JRC-Acquis (Steinberger et al., 2006) and DGT-TM (Steinberger et al., 2012). 2 data sets are out-of-domain, made available by the 2012 Workshop on Statistical Machine Translation (Callison-Burch et al., 2012). The development sets are random samples from the respective in-domain bitexts (held-out from training). The test sets have been provided by Translated, our industry partner in the MATECAT project.

Our second data set is CzEng 0.9, a Czech–English parallel corpus (Bojar and Zabokrtský, 2009). It contains text from 7 different sources, on which we train separate component models. The size of the corpora is shown in table 3. As development and test sets, we use 500 sentences of held-out data per source.

For both data sets, language models are trained on the target side of the bitexts. In all experiments, we keep the number of component models constant: 12 for EN–DE, 7 for CZ–EN. We vary the number of clusters  $k$  from 1, which corresponds to adapting the models to the full development set, to 16. The baseline is the concatenation of all train-

Data set	$\lambda_{IT}$	$\lambda_{LEGAL}$	$\lambda_{cluster 1}$	$\lambda_{cluster 2}$
kde	1.0	1.0	1.0	1.0
kdedoc	0.64	12.0	86.0	6.4
kdegb	1.6	2.3	1.7	2.7
oo	0.76	1.6	0.73	1.7
oo3	1.8	4.7	2.4	2.7
php	0.79	6.3	0.69	3.5
tm	1.3	1.3	1.5	1.1
acquis	0.024	3.5	0.018	1.9
dgt	0.053	4.5	0.033	2.4
ecb	0.071	2.3	0.039	1.2
ep7	0.037	0.53	0.024	0.29
nc7	0.1	1.1	0.063	0.62

Table 5: Weight vectors for feature  $p(\bar{t}|\bar{s})$  optimized on four development sets (from gold split and clustering with  $k = 2$ ).

ing data, with no adaptation performed. We also evaluate the labelled setting, where instead of unsupervised clustering, we use gold labels to split the development and test sets, and adapt the models to each labelled domain.

## 5.2 Results

Table 4 shows results for the EN–DE data set. For our clustering experiments, the development set is the concatenation of the LEGAL and IT development sets. However, we always use the gold segmentation between LEGAL and IT for MERT and testing. This allows for a detailed analysis of the effect of development data clustering for the purpose of model adaptation. In an unlabelled setting, one would have to run MERT either on the full development set (as we will do for the CZ–EN task) or separately on each cluster, or use an alternative approach to optimize log-linear weights in a multi-domain setting, such as feature augmentation as described by (Clark et al., 2012).

system	TM adaptation	LM adaptation	TM+LM adaptation
baseline	34.4	34.4	34.4
1 cluster (no split)	34.5	33.7	34.1
2 clusters	34.6	34.0	34.4
4 clusters	34.7*	34.3	34.6
8 clusters	34.7*	34.5	34.9*
16 clusters	34.7*	34.7*	35.0*
gold clusters	35.0*	35.0*	35.4*

Table 6: Translation experiments CZ–EN. BLEU scores reported.

We find that an adaptation of the TM and LM to the full development set (system “1 cluster”) yields the smallest improvements over the unadapted baseline. The reason for this is that the mixed-domain development set is not representative for the respective test sets. Using multiple adapted systems yields better performance. For the IT test set, the system with gold labels and TM adaptation yields an improvement of 0.7 BLEU (21.1  $\rightarrow$  21.8), LM adaptation yields 1.3 BLEU (21.1  $\rightarrow$  22.4), and adapting both models outperforms the baseline by 2.1 BLEU (21.1  $\rightarrow$  23.2). The systems that use unsupervised clusters reach a similar level of performance than those with gold clusters, with best results being achieved by the systems with 2–8 clusters. Some systems outperform both the baseline and the gold clusters, e.g. TM adaptation with 8 clusters (21.1  $\rightarrow$  21.8  $\rightarrow$  22.1), or LM adaptation with 4 or 8 clusters (21.1  $\rightarrow$  22.4  $\rightarrow$  23.1).

Results with 16 clusters are slightly worse than those with 2–8 clusters due to two effects. Firstly, for the system with adapted TM, one of the three MERT runs is an outlier, and the reported BLEU score of 21.1 is averaged from the three MERT runs achieving 22.1, 21.6, and 19.6 BLEU, respectively. Secondly, about one third of the IT test set is assigned to a cluster that is not IT-specific, which weakens the effect of domain adaptation for the systems with 16 clusters.

For the LEGAL subset, gains are smaller. This can be explained by the fact that the majority of training data is already from the legal domain, which makes it unnecessary to boost its impact on the probability distribution even further.

Table 5 shows the automatically obtained translation model weight vectors for two systems, “gold clusters” and “2 clusters”, for the feature  $p(\bar{t}|\bar{s})$ . It illustrates that all the corpora that we consider out-of-domain for IT are penalized by

a factor of 10–50 (relative to the in-domain *kde* corpus) for the computation of this feature. For the LEGAL domain, the weights are more uniform, which is congruent with our observation that BLEU changes little.

Table 6 shows results for the CZ–EN data set. For each system, MERT is performed on the full development set. As in the first task, adaptation to the full development set is least effective. The systems with unsupervised clusters significantly outperform the baseline. For the system with 16 clusters, we observe an improvement of 0.3 BLEU for TM adaptation, and 0.6 BLEU for adapting both models (34.4  $\rightarrow$  34.7  $\rightarrow$  35.0). The labelled system, i.e. the system with 7 clusters corresponding to the 7 data sources, both for the development and test set, performs best. We observe gains of 0.6 BLEU (34.4  $\rightarrow$  35.0) for TM or LM adaptation, and 1 BLEU (34.4  $\rightarrow$  35.4) when both models are adapted.

We conclude that the translation model architecture is effective in a multi-domain setting, both with unsupervised clusters and labelled domains. The fact that language model adaptation yields an additional improvement in our experiments suggests that it would be worthwhile to also investigate a language model data structure that efficiently supports multiple domains.

## 6 Conclusion

We have presented a novel translation model architecture that delays the computation of translation model features to the decoding phase, and uses a vector of component models for this computation. We have also described a usage scenario for this architecture, namely its ability to quickly switch between weight vectors in order to serve as an adapted model for multiple domains. A simple, unsupervised clustering of development data is sufficient to make use of this ability and imple-

ment a multi-domain translation system. If available, one can also use the architecture in a labelled setting.

Future work could involve merging our translation model framework with the online adaptation of other models, or the log-linear weights. Our approach is orthogonal to that of (Clark et al., 2012), who perform feature augmentation to obtain multiple sets of adapted log-linear weights. While (Clark et al., 2012) use labelled data, their approach could in principle also be applied after unsupervised clustering.

The translation model framework could also serve as the basis of real-time adaptation of translation systems, e.g. by using incremental means to update the weight vector, or having an incrementally trainable component model that learns from the post-edits by the user, and is assigned a suitable weight.

## Acknowledgments

This research was partially funded by the Swiss National Science Foundation under grant 105215\_126999, the European Commission (MATECAT, ICT-2011.4.2 287688) and the DARPA BOLT project.

## References

- Pratyush Banerjee, Jinhua Du, Baoli Li, Sudip Kumar Naskar, Andy Way, and Josef Van Genabith. 2010. Combining multi-domain statistical machine translation models using automatic classifiers. In *9th Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, Denver, Colorado, USA.
- Ondrej Bojar and Zdenek Zabokrtský. 2009. Czeg 0.9: Large parallel treebank with rich annotation. *Prague Bull. Math. Linguistics*, 92:63–84.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, June. Association for Computational Linguistics.
- Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Jonathan H. Clark, Alon Lavie, and Chris Dyer. 2012. One system, many domains: Open-domain statistical machine translation via feature augmentation. In *Conference of the Association for Machine Translation in the Americas 2012 (AMTA 2012)*, San Diego, California, USA.
- George Foster and Roland Kuhn. 2007. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation, StatMT '07*, pages 128–135, Prague, Czech Republic. Association for Computational Linguistics.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, Edmonton, Canada. Association for Computational Linguistics.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.
- Spyros Matsoukas, Antti-Veikko I. Rosti, and Bing Zhang. 2009. Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, pages 708–717, Singapore. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. 2010. Online learning for interactive statistical machine translation. In *HLT-NAACL*, pages 546–554. The Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Majid Razmara, George Foster, Baskaran Sankaran, and Anoop Sarkar. 2012. Mixing multiple translation models in statistical machine translation. In

- Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju, Republic of Korea. Association for Computational Linguistics.
- Rico Sennrich. 2012a. Mixture-modeling with unsupervised clusters for domain adaptation in statistical machine translation. In *16th Annual Conference of the European Association for Machine Translation (EAMT 2012)*, pages 185–192, Trento, Italy.
- Rico Sennrich. 2012b. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 539–549, Avignon, France. Association for Computational Linguistics.
- Kashif Shah, Loc Barrault, and Holger Schwenk. 2012. A general framework to weight heterogeneous parallel data for model adaptation in statistical machine translation. In *Conference of the Association for Machine Translation in the Americas 2012 (AMTA 2012)*, San Diego, California, USA.
- Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Daniel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006)*, Genoa, Italy.
- Ralf Steinberger, Andreas Eisele, Szymon Kloczek, Spyridon Pilos, and Patrick Schlüter. 2012. DGT-TM: A freely available translation memory in 22 languages. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).
- Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. In *Seventh International Conference on Spoken Language Processing*, pages 901–904, Denver, CO, USA.
- Jörg Tiedemann. 2009. News from OPUS - a collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.
- Hirofumi Yamamoto and Eiichiro Sumita. 2007. Bilingual cluster based models for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 514–523, Prague, Czech Republic.

# Part-of-Speech Induction in Dependency Trees for Statistical Machine Translation

Akihiro Tamura<sup>†,‡</sup>, Taro Watanabe<sup>†</sup>, Eiichiro Sumita<sup>†</sup>,  
Hiroya Takamura<sup>‡</sup>, Manabu Okumura<sup>‡</sup>

<sup>†</sup> National Institute of Information and Communications Technology  
{akihiro.tamura, taro.watanabe, eiichiro.sumita}@nict.go.jp

<sup>‡</sup> Precision and Intelligence Laboratory, Tokyo Institute of Technology  
{takamura, oku}@pi.titech.ac.jp

## Abstract

This paper proposes a nonparametric Bayesian method for inducing Part-of-Speech (POS) tags in dependency trees to improve the performance of statistical machine translation (SMT). In particular, we extend the monolingual infinite tree model (Finkel et al., 2007) to a bilingual scenario: each hidden state (POS tag) of a source-side dependency tree emits a source word together with its aligned target word, either jointly (joint model), or independently (independent model). Evaluations of Japanese-to-English translation on the NTCIR-9 data show that our induced Japanese POS tags for dependency trees improve the performance of a forest-to-string SMT system. Our independent model gains over 1 point in BLEU by resolving the sparseness problem introduced in the joint model.

## 1 Introduction

In recent years, syntax-based SMT has made promising progress by employing either dependency parsing (Lin, 2004; Ding and Palmer, 2005; Quirk et al., 2005; Shen et al., 2008; Mi and Liu, 2010) or constituency parsing (Huang et al., 2006; Liu et al., 2006; Galley et al., 2006; Mi and Huang, 2008; Zhang et al., 2008; Cohn and Blunsom, 2009; Liu et al., 2009; Mi and Liu, 2010; Zhang et al., 2011) on the source side, the target side, or both. However, dependency parsing, which is a popular choice for Japanese, can incorporate only shallow syntactic information, i.e., POS tags, compared with the richer syntactic phrasal categories in constituency parsing. Moreover, existing POS tagsets might not be optimal for SMT because they are constructed without considering the language in the other side. Consider the examples in Figure 1. The Japanese noun “利用” in

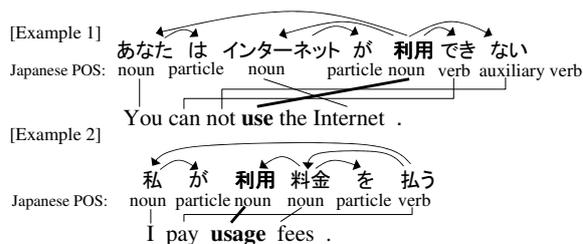


Figure 1: Examples of Existing Japanese POS Tags and Dependency Structures

Example 1 corresponds to the English verb “use”, while that in Example 2 corresponds to the English noun “usage”. Thus, Japanese nouns act like verbs in English in one situation, and nouns in English in another. If we could discriminate POS tags for two cases, we might improve the performance of a Japanese-to-English SMT system.

In the face of the above situations, this paper proposes an unsupervised method for inducing POS tags for SMT, and aims to improve the performance of syntax-based SMT by utilizing the induced POS tagset. The proposed method is based on the infinite tree model proposed by Finkel et al. (2007), which is a nonparametric Bayesian method for inducing POS tags from syntactic dependency structures. In this model, hidden states represent POS tags, the observations they generate represent the words themselves, and tree structures represent syntactic dependencies between pairs of POS tags.

The proposed method builds on this model by incorporating the aligned words in the other language into the observations. We investigate two types of models: (i) a joint model and (ii) an independent model. In the joint model, each hidden state jointly emits both a source word and its aligned target word as an observation. The independent model separately emits words in two languages from hidden states. By inferring POS

tags based on bilingual observations, both models can induce POS tags by incorporating information from the other language. Consider, for example, inducing a POS tag for the Japanese word “利用” in Figure 1. Under a monolingual induction method (e.g., the infinite tree model), the “利用” in Example 1 and 2 would both be assigned the same POS tag since they share the same observation. However, our models would assign separate tags for the two different instances since the “利用” in Example 1 and Example 2 could be disambiguated by encoding the target-side information, either “use” or “usage”, in the observations.

Inference is efficiently carried out by beam sampling (Gael et al., 2008), which combines slice sampling and dynamic programming. Experiments are carried out on the NTCIR-9 Japanese-to-English task using a binarized forest-to-string SMT system with dependency trees as its source side. Our bilingually-induced tagset significantly outperforms the original tagset and the monolingually-induced tagset. Further, our independent model achieves a more than 1 point gain in BLEU, which resolves the sparseness problem introduced by the bi-word observations.

## 2 Related Work

A number of unsupervised methods have been proposed for inducing POS tags. Early methods have the problem that the number of possible POS tags must be provided preliminarily. This limitation has been overcome by automatically adjusting the number of possible POS tags using non-parametric Bayesian methods (Finkel et al., 2007; Gael et al., 2009; Blunsom and Cohn, 2011; Sirts and Alumäe, 2012). Gael et al. (2009) applied infinite HMM (iHMM) (Beal et al., 2001; Teh et al., 2006), a nonparametric version of HMM, to POS induction. Blunsom and Cohn (2011) used a hierarchical Pitman-Yor process prior to the transition and emission distribution for sophisticated smoothing. Sirts and Alumäe (2012) built a model that combines POS induction and morphological segmentation into a single learning problem. Finkel et al. (2007) proposed the infinite tree model, which represents recursive branching structures over infinite hidden states and induces POS tags from syntactic dependency structures. In the following, we overview the infinite tree model, which is the basis of our proposed model. In particular, we will describe the independent children

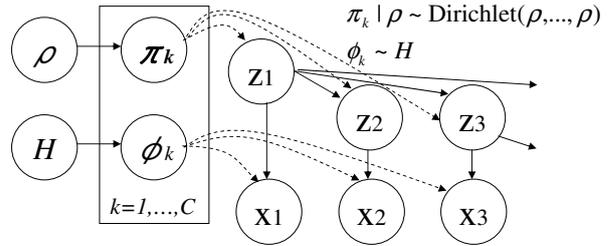


Figure 2: A Graphical Representation of the Finite Tree Model

model (Finkel et al., 2007), where children are dependent only on their parents, used in our proposed model<sup>1</sup>.

### 2.1 Finite Tree Model

We first review the finite tree model, which can be graphically represented in Figure 2. Let  $T_t$  denote the tree whose root node is  $t$ . A node  $t$  has a hidden state  $z_t$  (the POS tag) and an observation  $x_t$  (the word). The probability of a tree  $T_t$ ,  $p_T(T_t)$ , is recursively defined:  $p_T(T_t) = p(x_t|z_t) \prod_{t' \in c(t)} p(z_{t'}|z_t) p_T(T_{t'})$ ,

where  $c(t)$  is the set of the children of  $t$ .

Let each hidden state variable have  $C$  possible values indexed by  $k$ . For each state  $k$ , there is a parameter  $\phi_k$  which parameterizes the observation distribution for that state:  $x_t|z_t \sim F(\phi_{z_t})$ .  $\phi_k$  is distributed according to a prior distribution  $H$ :  $\phi_k \sim H$ .

Transitions between states are governed by Markov dynamics parameterized by  $\pi$ , where  $\pi_{ij} = p(z_{c(t)} = j|z_t = i)$  and  $\pi_k$  are the transition probabilities from the parent’s state  $k$ .  $\pi_k$  is distributed according to a Dirichlet distribution with parameter  $\rho$ :  $\pi_k|\rho \sim \text{Dirichlet}(\rho, \dots, \rho)$ . The hidden state of each child  $z_{t'}$  is distributed according to a multinomial distribution  $\pi_{z_t}$  specific to the parent’s state  $z_t$ :  $z_{t'}|z_t \sim \text{Multinomial}(\pi_{z_t})$ .

### 2.2 Infinite Tree Model

In the infinite tree model, the number of possible hidden states is potentially infinite. The infinite model is formed by extending the finite tree model using a hierarchical Dirichlet process (HDP) (Teh et al., 2006). The reason for using an HDP rather

<sup>1</sup>Finkel et al. (2007) originally proposed three types of models: besides the independent children model, the simultaneous children model and the markov children model. Although we could apply the other two models, we leave this for future work.

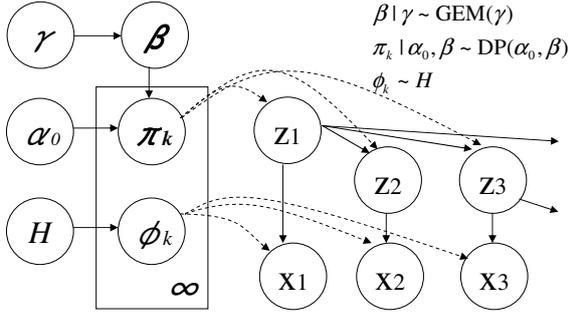


Figure 3: A Graphical Representation of the Infinite Tree Model

than a simple Dirichlet process (DP)<sup>2</sup> (Ferguson, 1973) is that we have to introduce coupling across transitions from different parent’s states. A similar measure was adopted in iHMM (Beal et al., 2001).

HDP is a set of DPs coupled through a shared random base measure which is itself drawn from a DP: each  $G_k \sim \text{DP}(\alpha_0, G_0)$  with a shared base measure  $G_0$ , and  $G_0 \sim \text{DP}(\gamma, H)$  with a global base measure  $H$ . From the viewpoint of the stick-breaking construction<sup>3</sup> (Sethuraman, 1994), the

HDP is interpreted as follows:  $G_0 = \sum_{k'=1}^{\infty} \beta_{k'} \delta_{\phi_{k'}}$

and  $G_k = \sum_{k'=1}^{\infty} \pi_{kk'} \delta_{\phi_{k'}}$ , where  $\beta \sim \text{GEM}(\gamma)$ ,  $\pi_k \sim \text{DP}(\alpha_0, \beta)$ , and  $\phi_{k'} \sim H$ .

We regard each  $G_k$  as two coindexed distributions:  $\pi_k$ , a distribution over the transition probabilities from the parent’s state  $k$ , and  $\phi_{k'}$ , an observation distribution for the state  $k'$ . Then, the infinite tree model is formally defined as follows:

$$\begin{aligned} \beta | \gamma &\sim \text{GEM}(\gamma), \\ \pi_k | \alpha_0, \beta &\sim \text{DP}(\alpha_0, \beta), \\ \phi_k &\sim H, \\ z_t | z_t &\sim \text{Multinomial}(\pi_{z_t}), \\ x_t | z_t &\sim F(\phi_{z_t}). \end{aligned}$$

Figure 3 shows the graphical representation of the infinite tree model. The primary difference be-

<sup>2</sup>DP is a measure on measures. It has two parameters, a scaling parameter  $\alpha$  and a base measure  $H$ :  $\text{DP}(\alpha, H)$ .

<sup>3</sup>Sethuraman (1994) showed a definition of a measure  $G \sim \text{DP}(\alpha_0, G_0)$ . First, infinite sequences of i.i.d variables  $(\pi'_k)_{k=1}^{\infty}$  and  $(\phi_k)_{k=1}^{\infty}$  are generated:  $\pi'_k | \alpha_0 \sim \text{Beta}(1, \alpha_0)$ ,  $\phi_k \sim G_0$ . Then,  $G$  is defined as:  $\pi_k = \pi'_k \sum_{l=1}^{k-1} (1 - \pi'_l)$ ,  $G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$ . If  $\pi$  is defined by this process, then we write  $\pi \sim \text{GEM}(\alpha_0)$ .

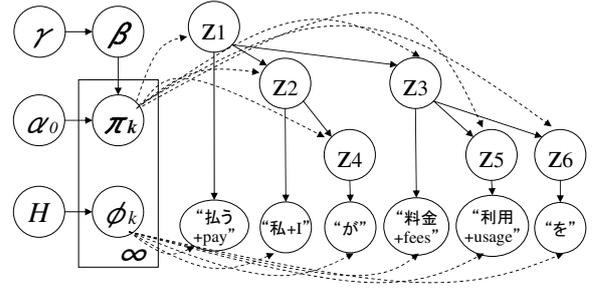


Figure 4: An Example of the Joint Model

tween Figure 2 and Figure 3 is whether the number of copies of the state is finite or not.

### 3 Bilingual Infinite Tree Model

We propose a bilingual variant of the infinite tree model, the bilingual infinite tree model, which utilizes information from the other language. Specifically, the proposed model introduces bilingual observations by embedding the aligned target words in the source-side dependency trees. This paper proposes two types of models that differ in their processes for generating observations: the joint model and the independent model.

#### 3.1 Joint Model

The joint model is a simple application of the infinite tree model under a bilingual scenario. The model is formally defined in the same way as in Section 2.2 and is graphically represented similarly to Figure 3. The only difference from the infinite tree model is the instances of observations ( $x_t$ ). Observations in the joint model are the combination of source words and their aligned target words<sup>4</sup>, while observations in the monolingual infinite tree model represent only source words. For each source word, all the aligned target words are copied and sorted in alphabetical order, and then concatenated into a single observation. Therefore, a single target word may be emitted multiple times if the target word is aligned with multiple source words. Likewise, there may be target words which may not be emitted by our model, if the target words are not aligned.

Figure 4 shows the process of generating Example 2 in Figure 1 through the joint model, where aligned words are jointly emitted as observations. In Figure 4, the POS tag of “利用” ( $z_5$ ) generates

<sup>4</sup>When no target words are aligned, we simply add a NULL target word.

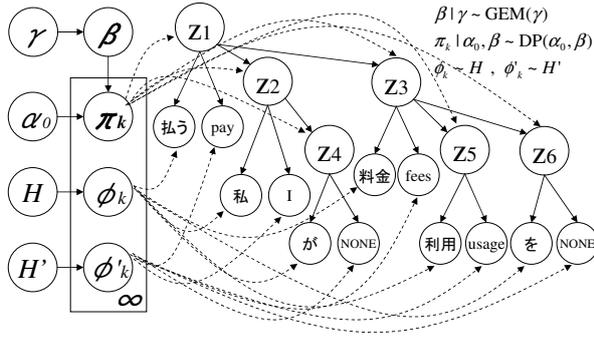


Figure 5: A Graphical Representation of the Independent Model

the string “利用+usage” as the observation ( $x_5$ ). Similarly, the POS tag of “利用” in Example 1 would generate the string “利用+use”. Hence, this model can assign different POS tags to the two different instances of the word “利用”, based on the different observation distributions in inference.

### 3.2 Independent Model

The joint model is prone to a data sparseness problem, since each observation is a combination of a source word and its aligned target word. Thus, we propose an independent model, where each hidden state generates a source word and its aligned target word separately. For the aligned target side, we introduce an observation variable  $x'_t$  for each  $z_t$  and a parameter  $\phi'_k$  for each state  $k$ , which parameterizes a distinct distribution over the observations  $x'_t$  for that state.  $\phi'_k$  is distributed according to a prior distribution  $H'$ . Specifically, the independent model is formally defined as follows:

$$\begin{aligned} \beta | \gamma &\sim \text{GEM}(\gamma), \\ \pi_k | \alpha_0, \beta &\sim \text{DP}(\alpha_0, \beta), \\ \phi_k &\sim H, \quad \phi'_k \sim H', \\ z_{t'} | z_t &\sim \text{Multinomial}(\pi_{z_t}), \\ x_t | z_t &\sim F(\phi_{z_t}), \quad x'_t | z_t \sim F'(\phi'_{z_t}). \end{aligned}$$

When multiple target words are aligned to a single source word, each aligned word is generated separately from observation distribution parameterized by  $\phi'_k$ .

Figure 5 graphs the process of generating Example 2 in Figure 1 using the independent model.  $x'_t$  and  $\phi'_k$  are introduced for aligned target words. The state of “利用” ( $z_5$ ) generates the Japanese word “利用” as  $x_5$  and the English word “usage” as  $x'_5$ . Due to this factorization, the independent model is less subject to the sparseness problem.

### 3.3 Introduction of Other Factors

We assumed the surface form of aligned target words as additional observations in previous sections. Here, we introduce additional factors, i.e., the POS of aligned target words, in the observations. Note that POSs of target words are assigned by a POS tagger in the target language and are not inferred in the proposed model.

First, we can simply replace surface forms of target words with their POSs to overcome the sparseness problem. Second, we can incorporate both information from the target language as observations. In the joint model, two pieces of information are concatenated into a single observation. In the independent model, we introduce observation variables (e.g.,  $x'_t$  and  $x''_t$ ) and parameters (e.g.,  $\phi'_k$  and  $\phi''_k$ ) for each information. Specifically,  $x'_t$  and  $\phi'_k$  are introduced for the surface form of aligned words, and  $x''_t$  and  $\phi''_k$  for the POS of aligned words. Consider, for example, Example 1 in Figure 1. The POS tag of “利用” generates the string “利用+use+verb” as the observation in the joint model, while it generates “利用”, “use”, and “verb” independently in the independent model.

### 3.4 POS Refinement

We have assumed a completely unsupervised way of inducing POS tags in dependency trees. Another realistic scenario is to refine the existing POS tags (Finkel et al., 2007; Liang et al., 2007) so that each refined sub-POS tag may reflect the information from the aligned words while preserving the handcrafted distinction from original POS tagset. Major difference is that we introduce separate transition probabilities  $\pi_k^s$  and observation distributions ( $\phi_k^s, \phi'_k{}^s$ ) for each existing POS tag  $s$ . Then, each node  $t$  is constrained to follow the distributions indicated by the initially assigned POS tag  $s_t$ , and we use the pair  $(s_t, z_t)$  as a state representation.

### 3.5 Inference

In inference, we find the state set that maximizes the posterior probability of state transitions given observations (i.e.,  $P(z_{1:n} | x_{1:n})$ ). However, we cannot evaluate the probability for all possible states because the number of states is infinite. Finkel et al. (2007) presented a sampling algorithm for the infinite tree model, which is based on the Gibbs sampling in the direct assignment representation for iHMM (Teh et al., 2006). In the

Gibbs sampling, individual hidden state variables are resampled conditioned on all other variables. Unfortunately, its convergence is slow in HMM settings because sequential data is likely to have a strong correlation between hidden states (Gael et al., 2008).

We present an inference procedure based on beam sampling (Gael et al., 2008) for the joint model and the independent model. Beam sampling limits the number of possible state transitions for each node to a finite number using slice sampling (Neal, 2003), and then efficiently samples whole hidden state transitions using dynamic programming. Beam sampling does not suffer from slow convergence as in Gibbs sampling by sampling the whole state variables at once. In addition, Gael et al. (2008) showed that beam sampling is more robust to initialization and hyperparameter choice than Gibbs sampling.

Specifically, we introduce an auxiliary variable  $u_t$  for each node in a dependency tree to limit the number of possible transitions. Our procedure alternates between sampling each of the following variables: the auxiliary variables  $\mathbf{u}$ , the state assignments  $\mathbf{z}$ , the transition probabilities  $\boldsymbol{\pi}$ , the shared DP parameters  $\boldsymbol{\beta}$ , and the hyperparameters  $\alpha_0$  and  $\gamma$ . We can parallelize procedures in sampling  $\mathbf{u}$  and  $\mathbf{z}$  because the slice sampling for  $\mathbf{u}$  and the dynamic programming for  $\mathbf{z}$  are independent for each sentence. See Gael et al. (2009) for details.

The only difference between inferences in the joint model and the independent model is in computing the posterior probability of state transitions given observations (e.g.,  $p(z_{1:n}|x_{1:n})$  and  $p(z_{1:n}|x_{1:n}, x'_{1:n})$ ) in sampling  $\mathbf{z}$ . In the following, we describe each sampling stage. See Teh et al., (2006) for details of sampling  $\boldsymbol{\pi}$ ,  $\boldsymbol{\beta}$ ,  $\alpha_0$  and  $\gamma$ .

### Sampling $\mathbf{u}$ :

Each  $u_t$  is sampled from the uniform distribution on  $[0, \pi_{z_{d(t)}z_t}]$ , where  $d(t)$  is the parent of  $t$ :  $u_t \sim \text{Uniform}(0, \pi_{z_{d(t)}z_t})$ . Note that  $u_t$  is a positive number, since each transition probability  $\pi_{z_{d(t)}z_t}$  is larger than zero.

### Sampling $\mathbf{z}$ :

Possible values  $k$  of  $z_t$  are divided into the two sets using  $u_t$ : a finite set with  $\pi_{z_{d(t)}k} > u_t$  and an infinite set with  $\pi_{z_{d(t)}k} \leq u_t$ . The beam sampling considers only the former set. Owing to the truncation of the latter set, we can compute the posterior probability of a state  $z_t$  given ob-

servations for all  $t$  ( $t = 1, \dots, T$ ) using dynamic programming as follows:

In the joint model,  $p(z_t|x_{\sigma(t)}, u_{\sigma(t)}) \propto p(x_t|z_t) \cdot \sum_{z_{d(t)}: \pi_{z_{d(t)}z_t} > u_t} p(z_{d(t)}|x_{\sigma(d(t))}, u_{\sigma(d(t))})$ ,

and in the independent model,  $p(z_t|x_{\sigma(t)}, x'_{\sigma(t)}, u_{\sigma(t)}) \propto p(x_t|z_t) \cdot p(x'_t|z_t) \cdot \sum_{z_{d(t)}: \pi_{z_{d(t)}z_t} > u_t} p(z_{d(t)}|x_{\sigma(d(t))}, x'_{\sigma(d(t))}, u_{\sigma(d(t))})$ ,

where  $x_{\sigma(t)}$  (or  $u_{\sigma(t)}$ ) denotes the set of  $x_t$  (or  $u_t$ ) on the path from the root node to the node  $t$  in a tree.

In our experiments, we assume that  $F(\phi_k)$  is Multinomial( $\phi_k$ ) and  $H$  is Dirichlet( $\rho, \dots, \rho$ ), which is the same in Finkel et al. (2007). Under this assumption, the posterior probability of an observation is as follows:  $p(x_t|z_t) = \frac{\dot{n}_{x_tk} + \rho}{\dot{n}_{\cdot k} + N\rho}$ , where  $\dot{n}_{x_tk}$  is the number of observations  $x$  with state  $k$ ,  $\dot{n}_{\cdot k}$  is the number of hidden states whose values are  $k$ , and  $N$  is the total number of observations  $\mathbf{x}$ . Similarly,  $p(x'_t|z_t) = \frac{\dot{n}'_{x'_tk} + \rho'}{\dot{n}'_{\cdot k} + N'\rho'}$ , where  $N'$  is the total number of observations  $\mathbf{x}'$ .

When the posterior probability of a state  $z_t$  given observations for all  $t$  can be computed, we first sample the state of each leaf node and then perform backtrack sampling for every other  $z_t$  where the  $z_t$  is sampled given the sample for  $z_{c(t)}$  as follows:  $p(z_t|z_{c(t)}, x_{1:T}, u_{1:T}) \propto p(z_t|x_{\sigma(t)}, u_{\sigma(t)}) \prod_{t' \in c(t)} p(z_{t'}|z_t, u_{t'})$ .

### Sampling $\boldsymbol{\pi}$ :

We introduce a count variable  $n_{ij} \in \mathbf{n}$ , which is the number of observations with state  $j$  whose parent's state is  $i$ . Then, we sample  $\boldsymbol{\pi}$  using the Dirichlet distribution:  $(\pi_{k1}, \dots, \pi_{kK}, \sum_{k'=K+1}^{\infty} \pi_{kk'}) \sim \text{Dirichlet}(n_{k1} + \alpha_0\beta_1, \dots, n_{kK} + \alpha_0\beta_K, \alpha_0 \sum_{k'=K+1}^{\infty} \beta_{k'})$ , where  $K$  is the number of distinct states in  $\mathbf{z}$ .

### Sampling $\boldsymbol{\beta}$ :

We introduce a set of auxiliary variables  $\mathbf{m}$ , where  $m_{ij} \in \mathbf{m}$  is the number of elements of  $\boldsymbol{\pi}_j$  corresponding to  $\beta_i$ . The conditional distribution of each variable is  $p(m_{ij} = m|z, \boldsymbol{\beta}, \alpha_0) \propto S(n_{ij}, m)(\alpha_0\beta_j)^m$ , where  $S(n, m)$  are unsigned Stirling numbers of the first kind<sup>5</sup>.

<sup>5</sup> $S(0, 0) = S(1, 1) = 1$ ,  $S(n, 0) = 0$  for  $n > 0$ ,  $S(n, m) = 0$  for  $m > n$ , and  $S(n + 1, m) = S(n, m - 1) + nS(n, m)$  for others.

The parameters  $\beta$  are sampled using the Dirichlet distribution:  $(\beta_1, \dots, \beta_K, \sum_{k'=K+1}^{\infty} \beta_{k'}) \sim \text{Dirichlet}(m_{\cdot 1}, \dots, m_{\cdot K}, \gamma)$ , where  $m_{\cdot k} = \sum_{k'=1}^K m_{k'k}$ .

### Sampling $\alpha_0$ :

$\alpha_0$  is parameterized by a gamma hyperprior with hyperparameters  $\alpha_a$  and  $\alpha_b$ . We introduce two types of auxiliary variables for each state ( $k = 1, \dots, K$ ),  $w_k \in [0, 1]$  and  $v_k \in \{0, 1\}$ . The conditional distribution of each  $w_k$  is  $p(w_k|\alpha_0) \propto w_k^{\alpha_0} (1-w_k)^{n_{\cdot k}-1}$  and that of each  $v_k$  is  $p(v_k|\alpha_0) \propto \left(\frac{n_{\cdot k}}{\alpha_0}\right)^{v_k}$ , where  $n_{\cdot k} = \sum_{k'=1}^K n_{k'k}$ . The conditional distribution of  $\alpha_0$  given  $w_k$  and  $v_k$  ( $k = 1, \dots, K$ ) is  $p(\alpha_0|\mathbf{w}, \mathbf{v}) \propto \alpha_0^{\alpha_a-1+m_{\cdot\cdot}-\sum_{k=1}^K v_k} e^{-\alpha_0(\alpha_b-\sum_{k=1}^K \log w_k)}$ , where  $m_{\cdot\cdot} = \sum_{k'=1}^K \sum_{k''=1}^K m_{k'k''}$ .

### Sampling $\gamma$ :

$\gamma$  is parameterized by a gamma hyperprior with hyperparameters  $\gamma_a$  and  $\gamma_b$ . We introduce an auxiliary variable  $\eta$ , whose conditional distribution is  $p(\eta|\gamma) \propto \eta^\gamma (1-\eta)^{m_{\cdot\cdot}-1}$ . The conditional distribution of  $\gamma$  given  $\eta$  is  $p(\gamma|\eta) \propto \gamma^{\gamma_a-1+K} e^{-\gamma(\gamma_b-\log \eta)}$ .

## 4 Experiment

We tested our proposed models under the NTCIR-9 Japanese-to-English patent translation task (Goto et al., 2011), consisting of approximately 3.2 million bilingual sentences. Both the development data and the test data consist of 2,000 sentences. We also used the NTCIR-7 development data consisting of 2,741 sentences for development testing purposes.

### 4.1 Experimental Setup

We evaluated our bilingual infinite tree model for POS induction using an in-house developed syntax-based forest-to-string SMT system. In the training process, the following steps are performed sequentially: preprocessing, inducing a POS tagset for a source language, training a POS tagger and a dependency parser, and training a forest-to-string MT model.

#### Step 1. Preprocessing

We used the first 10,000 Japanese-English sentence pairs in the NTCIR-9 training data for in-

ducing a POS tagset for Japanese<sup>6</sup>. The Japanese sentences were segmented using MeCab<sup>7</sup>, and the English sentences were tokenized and POS tagged using TreeTagger (Schmid, 1994), where 43 and 58 types of POS tags are included in the Japanese sentences and the English sentences, respectively. The Japanese POS tags come from the second-level POS tags in the IPA POS tagset (Asahara and Matsumoto, 2003) and the English POS tags are derived from the Penn Treebank. Note that the Japanese POS tags are used for initialization of hidden states and the English POS tags are used as observations emitted by hidden states.

Word-by-word alignments for the sentence pairs are produced by first running GIZA++ (Och and Ney, 2003) in both directions and then combining the alignments using the “grow-diag-final-and” heuristic (Koehn et al., 2003). Note that we ran GIZA++ on all of the NTCIR-9 training data in order to obtain better alignments.

The Japanese sentences are parsed using CaboCha (Kudo and Matsumoto, 2002), which generates dependency structures using a phrasal unit called a *bunsetsu*<sup>8</sup>, rather than a word unit as in English or Chinese dependency parsing. Since we focus on the word-level POS induction, each *bunsetsu*-based dependency tree is converted into its corresponding word-based dependency tree using the following heuristic<sup>9</sup>: first, the last function word inside each *bunsetsu* is identified as the head word<sup>10</sup>; then, the remaining words are treated as dependents of the head word in the same *bunsetsu*; finally, a *bunsetsu*-based dependency structure is transformed to a word-based dependency structure by preserving the head/modifier relationships of the determined head words.

#### Step 2. POS Induction

A POS tag for each word in the Japanese sentences is inferred by our bilingual infinite tree model, ei-

<sup>6</sup>Due to the high computational cost, we did not use all the NTCIR-9 training data. We leave scaling up to a larger dataset for future work.

<sup>7</sup><http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html>

<sup>8</sup>A *bunsetsu* is the smallest meaningful sequence consisting of a content word and accompanying function words (e.g., a noun and a particle).

<sup>9</sup>We could use other word-based dependency trees such as trees by the infinite PCFG model (Liang et al., 2007) and syntactic-head or semantic-head dependency trees in Nakazawa and Kurohashi (2012), although it is not our major focus. We leave this for future work.

<sup>10</sup>If no function words exist in a *bunsetsu*, the last content word is treated as the head word.

ther jointly (*Joint*) or independently (*Ind*). We also performed monolingual induction of Finkel et al. (2007) for comparison (*Mono*). In each model, a sequence of sampling  $u$ ,  $z$ ,  $\pi$ ,  $\beta$ ,  $\alpha_0$ , and  $\gamma$  is repeated 10,000 times. In sampling  $\alpha_0$  and  $\gamma$ , hyperparameters  $\alpha_a$ ,  $\alpha_b$ ,  $\gamma_a$ , and  $\gamma_b$  are set to 2, 1, 1, and 1, respectively, which is the same setting in Gael et al. (2008). In sampling  $z$ , parameters  $\rho$ ,  $\rho'$ ,  $\dots$ , are set to 0.01. In the experiments, three types of factors for the aligned English words are compared: surface forms ('s'), POS tags ('P'), and the combination of both ('s+P'). Further, two types of inference frameworks are compared: *induction* (*IND*) and *refinement* (*REF*). In both frameworks, each hidden state  $z_t$  is first initialized to the POS tags assigned by MeCab (the IPA POS tagset), and then each state is updated through the inference procedure described in Section 3.5. Note that in *REF*, the sampling distribution over  $z_t$  is constrained to include only states that are a refinement of the initially assigned POS tag.

### Step 3. Training a POS Tagger and a Dependency Parser

In this step, we train a Japanese dependency parser from the 10,000 Japanese dependency trees with the induced POS tags which are derived from Step 2. We employed a transition-based dependency parser which can jointly learn POS tagging and dependency parsing (Hatori et al., 2011) under an incremental framework<sup>11</sup>. Note that the learned parser can identify dependencies between words and attach an induced POS tag for each word.

### Step 4. Training a Forest-to-String MT

In this step, we train a forest-to-string MT model based on the learned dependency parser in Step 3. We use an in-house developed hypergraph-based toolkit, *cicada*, for training and decoding with a tree-to-string model, which has been successfully employed in our previous work for system combination (Watanabe and Sumita, 2011) and online learning (Watanabe, 2012). All the Japanese and English sentences in the NTCIR-9 training data are segmented in the same way as in Step 1, and then each Japanese sentence is parsed by the dependency parser learned in Step 3, which simultaneously assigns induced POS tags and word dependencies. Finally, a forest-to-string MT model is learned with Zhang et al., (2011), which extracts translation rules by a forest-based variant of

<sup>11</sup><http://triple.cc/software/corbit/>

	<i>IND</i>	<i>REF</i>
<i>BS</i>	27.54	
<i>Mono</i>	27.66	26.83
<i>Joint</i> [s]	<b>28.00</b>	<b>28.00</b>
<i>Joint</i> [P]	26.36	26.72
<i>Joint</i> [s+P]	<b>27.99</b>	<b>27.82</b>
<i>Ind</i> [s]	<b>28.00</b>	<b>27.93</b>
<i>Ind</i> [P]	<b>28.11</b>	<b>28.63</b>
<i>Ind</i> [s+P]	<b>28.13</b>	<b>28.62</b>

Table 1: Performance on Japanese-to-English Translation Measured by BLEU (%)

the GHKM algorithm (Mi and Huang, 2008) after each parse tree is restructured into a binarized packed forest. Parameters are tuned on the development data using xBLEU (Rosti et al., 2011) as an objective and L-BFGS (Liu and Nocedal, 1989) as an optimization toolkit, since it is stable and less prone to randomness, unlike MERT (Och, 2003) or PRO (Hopkins and May, 2011). The development test data is used to set up hyperparameters, i.e., to terminate tuning iterations.

When translating Japanese sentences, a parse tree for each sentence is constructed in the same way as described earlier in this step, and then the parse trees are translated into English sentences using the learned forest-to-string MT model.

## 4.2 Experimental Results

Table 1 shows the performance for the test data measured by case sensitive BLEU (Papineni et al., 2002). We also present the performance of our baseline forest-to-string MT system (*BS*) using the original IPA POS tags. In Table 1, numbers in bold indicate that the systems outperform the baselines, *BS* and *Mono*. Under the Moses phrase-based SMT system (Koehn et al., 2007) with the default settings, we achieved a 26.80% BLEU score.

Table 1 shows that the proposed systems outperform the baseline *Mono*. The differences between the performance of *Ind*[s+P] and *Mono* are statistically significant in the bootstrap method (Koehn, 2004), with a 1% significance level both in *IND* and *REF*. The results indicate that integrating the aligned target-side information in POS induction makes inferred tagsets more suitable for SMT.

Table 1 also shows that the independent model is more effective for SMT than the joint model. This means that sparseness is a severe problem in

Model	<i>IND</i>	<i>REF</i>
<i>Joint</i> [s+P]	164	620
<i>Ind</i> [s+P]	102	517
IPA POS tags	42	

Table 2: The Number of POS Tags

POS induction when jointly encoding bilingual information into observations. Additionally, all the systems using the independent model outperform *BS*. The improvements are statistically significant in the bootstrap method (Koehn, 2004), with a 1% significance level. The results show that the proposed models can generate more favorable POS tagsets for SMT than an existing POS tagset.

In Table 1, *REF*s are at least comparable to, or better than, *IND*s except for *Mono*. This shows that *REF* achieves better performance by preserving the clues from the original POS tagset. However, *REF* may suffer severe overfitting problem for *Mono* since no bilingual information was incorporated. Further, when the full-level IPA POS tags<sup>12</sup> were used in *BS*, the system achieved a 27.49% BLEU score, which is worse than the result using the second-level IPA POS tags. This means that manual refinement without bilingual information may also cause an overfitting problem in MT.

## 5 Discussion

### 5.1 Comparison to the IPA POS Tagset

Table 2 shows the number of the IPA POS tags used in the experiments and the POS tags induced by the proposed models. This table shows that each induced tagset contains more POS tags than the IPA POS tagset. In the experimental data, some of Japanese verbs correspond to genuine English verbs, some are nominalized, and others correspond to English past participle verbs or present participle verbs which modify other words. Respective examples are “I use a card.”, “Using the index is faster.”, and “I explain using an example.”, where all the underlined words correspond to the same Japanese word, “用い”, whose IPA POS tag is a verb. *Ind*[s+P] in *REF* generated the POS tagset where the three types are assigned to separate POS groups.

The Japanese particle “に” is sometimes attached to nouns to give them adverb roles. For

<sup>12</sup>377 types of full-level IPA POS tags were included in our experimental data.

	Tagging		Dependency	
	<i>IND</i>	<i>REF</i>	<i>IND</i>	<i>REF</i>
<i>Original</i>	90.37		93.62	
<i>Mono</i>	90.75	88.04	91.77	91.51
<i>Joint</i> [s]	89.08	86.73	91.55	91.14
<i>Joint</i> [P]	80.54	79.98	91.06	91.29
<i>Joint</i> [s+P]	87.56	84.92	91.31	91.10
<i>Ind</i> [s]	87.62	84.33	92.06	92.58
<i>Ind</i> [P]	90.21	88.50	92.85	93.03
<i>Ind</i> [s+P]	89.57	86.12	92.96	92.78

Table 3: Tagging and Dependency Accuracy (%)

example, “相互 (mutual) に” is translated as the adverb “mutually” in English. Other times, it is attached to words to make them the objects of verbs. For example, “彼 (he) に 与える (give)” is translated as “give him”. The POS tags by *Ind*[s+P] in *REF* discriminated the two types.

These examples show that the proposed models can disambiguate POS tags that have different functions in English, whereas the IPA POS tagset treats them jointly. Thus, such discrimination improves the performance of a forest-to-string SMT.

### 5.2 Impact of Tagging and Dependency Accuracy

The performance of our methods depends not only on the quality of the induced tag sets but also on the performance of the dependency parser learned in Step 3 of Section 4.1. We cannot directly evaluate the tagging accuracy of the parser trained through Step 3 because we do not have any data with induced POS tags other than the 10,000-sentence data gained through Step 2. Thus we split the 10,000 data into the first 9,000 data for training and the remaining 1,000 for testing, and then a dependency parser was learned in the same way as in Step 3.

Table 3 shows the results. *Original* is the performance of the parser learned from the training data with the original POS tagset. Note that the dependency accuracies are measured on the automatically parsed dependency trees, not on the syntactically correct gold standard trees. Thus *Original* achieved the best dependency accuracy.

In Table 3, the performance for our bilingually-induced POSs, *Joint* and *Ind*, are lower than *Original* and *Mono*. It seems performing parsing and tagging with the bilingually-induced POS tagset is too difficult when only monolingual in-

formation is available to the parser. However, our bilingually-induced POSs, except for *Joint[P]*, with the lower accuracies are more effective for SMT than the monolingually-induced POSs and the original POSs, as indicated in Table 1. The tagging accuracies for *Joint[P]* both in *IND* and *REF* are significantly lower than the others, while the dependency accuracies do not differ significantly. The lower tagging accuracies may directly reflect the lower translation qualities for *Joint[P]* in Table 1.

## 6 Conclusion

We proposed a novel method for inducing POS tags for SMT. The proposed method is a non-parametric Bayesian method, which infers hidden states (i.e., POS tags) based on observations representing not only source words themselves but also aligned target words. Our experiments showed that a more favorable POS tagset can be induced by integrating aligned information, and furthermore, the POS tagset generated by the proposed method is more effective for SMT than an existing POS tagset (the IPA POS tagset).

Even though we employed word alignment from GIZA++ with potential errors, large gains were achieved using our proposed method. We would like to investigate the influence of alignment errors in the future. In addition, we are planning to prove the effectiveness of our proposed method for language pairs other than Japanese-to-English. We are also planning to introduce our proposed method to other syntax-based SMT, such as a string-to-tree SMT and a tree-to-tree SMT.

## Acknowledgments

We thank Isao Goto for helpful discussions and anonymous reviewers for valuable comments. We also thank Jun Hatori for helping us to apply his software, Corbit, to our induced POS tagsets.

## References

Masayuki Asahara and Yuji Matsumoto. 2003. IPADIC User Manual. Technical report, Japan.

Matthew J. Beal, Zoubin Ghahramani, and Carl E. Rasmussen. 2001. The Infinite Hidden Markov Model. In *Advances in Neural Information Processing Systems*, pages 577–584.

Phil Blunsom and Trevor Cohn. 2011. A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of

Speech Induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 865–874.

Trevor Cohn and Phil Blunsom. 2009. A Bayesian Model of Syntax-Directed Tree to String Grammar Induction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 352–361.

Yuan Ding and Martha Palmer. 2005. Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 541–548.

Thomas S. Ferguson. 1973. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):209–230.

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. 2007. The Infinite Tree. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 272–279.

Jurgen Van Gael, Yunus Saatci, Yee Whye Teh, and Zoubin Ghahramani. 2008. Beam Sampling for the Infinite Hidden Markov Model. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1088–1095.

Jurgen Van Gael, Andreas Vlachos, and Zoubin Ghahramani. 2009. The infinite HMM for unsupervised PoS tagging. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, pages 678–687.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable Inference and Training of Context-Rich Syntactic Translation Models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968.

Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the Patent Machine Translation Task at the NTCIR-9 Workshop. In *Proceedings of the 9th NTCIR Workshop*, pages 559–578.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental Joint POS Tagging and Dependency Parsing in Chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1216–1224.

Mark Hopkins and Jonathan May. 2011. Tuning as Ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. A Syntax-Directed Translator with Extended Domain of Locality. In *Proceedings of the Workshop on*

- Computationally Hard Problems and Joint Inference in Speech and Language Processing*, pages 1–8.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Human Language Technology Conference: North American Chapter of the Association for Computational Linguistics*, pages 48–54.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 177–180.
- Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395.
- Taku Kudo and Yuji Matsumoto. 2002. Japanese Dependency Analysis using Cascaded Chunking. In *Proceedings of the 6th Conference on Natural Language Learning*, pages 63–69.
- Percy Liang, Slav Petrov, Michael I. Jordan, and Dan Klein. 2007. The Infinite PCFG using Hierarchical Dirichlet Processes. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 688–697.
- Dekang Lin. 2004. A Path-based Transfer Model for Machine Translation. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 625–630.
- Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming B*, 45(3):503–528.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616.
- Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving Tree-to-Tree Translation with Packed Forests. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 558–566.
- Haitao Mi and Liang Huang. 2008. Forest-based Translation Rule Extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 206–214.
- Haitao Mi and Qun Liu. 2010. Constituency to Dependency Translation with Forests. In *Proceedings of the 48th Annual Conference of the Association for Computational Linguistics*, pages 1433–1442.
- Toshiaki Nakazawa and Sadao Kurohashi. 2012. Alignment by Bilingual Generation and Monolingual Derivation. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1963–1978.
- Radford M. Neal. 2003. Slice Sampling. *Annals of Statistics*, 31:705–767.
- Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29:19–51.
- Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics*, pages 271–279.
- Antti-Veikko Rosti, Bing Zhang, Spyros Matsoukas, and Richard Schwartz. 2011. Expected BLEU Training for Graphs: BBN System Description for WMT11 System Combination Task. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 159–165.
- Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49.
- Jayaram Sethuraman. 1994. A Constructive Definition of Dirichlet Priors. *Statistica Sinica*, 4(2):639–650.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model. In *Proceedings of the 46th Annual Conference of the Association for Computational Linguistics: Human Language Technologies*, pages 577–585.
- Kairit Sirts and Tanel Alumäe. 2012. A Hierarchical Dirichlet Process Model for Joint Part-of-Speech and Morphology Induction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 407–416.

- Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101(476):1566–1581.
- Taro Watanabe and Eiichiro Sumita. 2011. Machine Translation System Combination by Confusion Forest. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1249–1257.
- Taro Watanabe. 2012. Optimized Online Rank Learning for Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 253–262.
- Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A Tree Sequence Alignment-based Tree-to-Tree Translation Model. In *Proceedings of the 46th Annual Conference of the Association for Computational Linguistics: Human Language Technologies*, pages 559–567.
- Hao Zhang, Licheng Fang, Peng Xu, and Xiaoyun Wu. 2011. Binarized Forest to String Translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 19–24.

# Statistical Machine Translation Improves Question Retrieval in Community Question Answering via Matrix Factorization

Guangyou Zhou, Fang Liu, Yang Liu, Shizhu He, and Jun Zhao

National Laboratory of Pattern Recognition

Institute of Automation, Chinese Academy of Sciences

95 Zhongguancun East Road, Beijing 100190, China

{gyzhou, fliu, liuyang09, shizhu.he, jzhao}@nlpr.ia.ac.cn

## Abstract

Community question answering (CQA) has become an increasingly popular research topic. In this paper, we focus on the problem of question retrieval. Question retrieval in CQA can automatically find the most relevant and recent questions that have been solved by other users. However, the word ambiguity and word mismatch problems bring about new challenges for question retrieval in CQA. State-of-the-art approaches address these issues by implicitly expanding the queried questions with additional words or phrases using monolingual translation models. While useful, the effectiveness of these models is highly dependent on the availability of quality parallel monolingual corpora (e.g., question-answer pairs) in the absence of which they are troubled by noise issue. In this work, we propose an alternative way to address the word ambiguity and word mismatch problems by taking advantage of potentially rich semantic information drawn from other languages. Our proposed method employs statistical machine translation to improve question retrieval and enriches the question representation with the translated words from other languages via matrix factorization. Experiments conducted on a real CQA data show that our proposed approach is promising.

## 1 Introduction

With the development of Web 2.0, community question answering (CQA) services like Yahoo!

Answers,<sup>1</sup> Baidu Zhidao<sup>2</sup> and WkiAnswers<sup>3</sup> have attracted great attention from both academia and industry (Jeon et al., 2005; Xue et al., 2008; Adamic et al., 2008; Wang et al., 2009; Cao et al., 2010). In CQA, anyone can ask and answer questions on any topic, and people seeking information are connected to those who know the answers. As answers are usually explicitly provided by human, they can be helpful in answering real world questions.

In this paper, we focus on the task of question retrieval. Question retrieval in CQA can automatically find the most relevant and recent questions (historical questions) that have been solved by other users, and then the best answers of these historical questions will be used to answer the users' queried questions. However, question retrieval is challenging partly due to the **word ambiguity** and **word mismatch** between the queried questions and the historical questions in the archives. **Word ambiguity** often causes the retrieval models to retrieve many historical questions that do not match the users' intent. This problem is also amplified by the high diversity of questions and users. For example, depending on different users, the word "interest" may refer to "curiosity", or "a charge for borrowing money".

Another challenge is **word mismatch** between the queried questions and the historical questions. The queried questions may contain words that are different from, but related to, the words in the relevant historical questions. For example, if a queried question contains the word "company" but a relevant historical question instead contains the word "firm", then there is a mismatch and the historical

<sup>1</sup><http://answers.yahoo.com/>

<sup>2</sup><http://zhidao.baidu.com/>

<sup>3</sup><http://wiki.answers.com/>

	English	Chinese
word ambiguity	How do I get a loan from a <b>bank</b> ?	我(wǒ) 如何(rúhé) 从(cóng) 银行(yínháng) 贷款(dàikuǎn) ?
	How to reach the <b>bank</b> of the river?	如何(rúhé) 前往(qiánwǎng) 河岸(héàn) ?
word mismatch	company	公司(gōngsī)
	firm	公司(gōngsī)
	rheum	感冒(gǎnmào)
	catarrh	感冒(gǎnmào)

Table 1: Google translate: some illustrative examples.

question may not be easily distinguished from an irrelevant one.

Researchers have proposed the use of word-based translation models (Berger et al., 2000; Jeon et al., 2005; Xue et al., 2008; Lee et al., 2008; Bernhard and Gurevych, 2009) to solve the word mismatch problem. As a principle approach to capture semantic word relations, word-based translation models are built by using the IBM model 1 (Brown et al., 1993) and have been shown to outperform traditional models (e.g., VSM, BM25, LM) for question retrieval. Besides, Riezler et al. (2007) and Zhou et al. (2011) proposed the phrase-based translation models for question and answer retrieval. The basic idea is to capture the contextual information in modeling the translation of phrases as a whole, thus the word ambiguity problem is somewhat alleviated. However, all these existing studies in the literature are basically *monolingual approaches* which are restricted to the use of original language of questions. While useful, the effectiveness of these models is highly dependent on the availability of quality parallel monolingual corpora (e.g., question-answer pairs) in the absence of which they are troubled by noise issue. In this work, we propose an alternative way to address the word ambiguity and word mismatch problems by taking advantage of potentially rich semantic information drawn from other languages. Through other languages, various ways of adding semantic information to a question could be available, thereby leading to potentially more improvements than using the original language only.

Taking a step toward using other languages, we propose the use of *translated representation* by alternatively enriching the original questions with the words from other languages. The idea of improving question retrieval with statistical machine translation is based on the following two observa-

tions: (1) Contextual information is exploited during the translation from one language to another. For example in Table 1, English words “interest” and “bank” that have multiple meanings under different contexts are correctly addressed by using the state-of-the-art translation tool — **Google Translate**.<sup>4</sup> Thus, word ambiguity based on contextual information is naturally involved when questions are translated. (2) Multiple words that have similar meanings in one language may be translated into an unique word or a few words in a foreign language. For example in Table 1, English words such as “company” and “firm” are translated into “公司(gōngsī)”, “rheum” and “catarrh” are translated into “感冒(gǎnmào)” in Chinese. Thus, word mismatch problem can be somewhat alleviated by using other languages.

Although Zhou et al. (2012) exploited bilingual translation for question retrieval and obtained the better performance than traditional monolingual translation models. However, there are two problems with this enrichment: (1) enriching the original questions with the translated words from other languages increases the dimensionality and makes the question representation even more sparse; (2) statistical machine translation may introduce noise, which can harm the performance of question retrieval. To solve these two problems, we propose to leverage statistical machine translation to improve question retrieval via matrix factorization.

The remainder of this paper is organized as follows. Section 2 describes the proposed method by leveraging statistical machine translation to improve question retrieval via matrix factorization. Section 3 presents the experimental results. In section 4, we conclude with ideas for future research.

<sup>4</sup><http://translate.google.com/translate.t>

## 2 Our Approach

### 2.1 Problem Statement

This paper aims to leverage statistical machine translation to enrich the question representation. In order to address the word ambiguity and word mismatch problems, we expand a question by adding its translation counterparts. Statistical machine translation (e.g., Google Translate) can utilize contextual information during the question translation, so it can solve the word ambiguity and word mismatch problems to some extent.

Let  $L = \{l_1, l_2, \dots, l_P\}$  denote the language set, where  $P$  is the number of languages considered in the paper,  $l_1$  denotes the original language (e.g., English) while  $l_2$  to  $l_P$  are the foreign languages. Let  $D_1 = \{d_1^{(1)}, d_2^{(1)}, \dots, d_N^{(1)}\}$  be the set of historical question collection in original language, where  $N$  is the number of historical questions in  $D_1$  with vocabulary size  $M_1$ . Now we first translate each original historical question from language  $l_1$  into other languages  $l_p$  ( $p \in [2, P]$ ) by Google Translate. Thus, we can obtain  $D_2, \dots, D_P$  in different languages, and  $M_p$  is the vocabulary size of  $D_p$ . A question  $d_i^{(p)}$  in  $D_p$  is simply represented as a  $M_p$  dimensional vector  $\mathbf{d}_i^{(p)}$ , in which each entry is calculated by tf-idf. The  $N$  historical questions in  $D_p$  are then represented in a  $M_p \times N$  term-question matrix  $\mathbf{D}_p = \{\mathbf{d}_1^{(p)}, \mathbf{d}_2^{(p)}, \dots, \mathbf{d}_N^{(p)}\}$ , in which each row corresponds to a term and each column corresponds to a question.

Intuitively, we can enrich the original question representation by adding the translated words from language  $l_2$  to  $l_P$ , the original vocabulary size is increased from  $M_1$  to  $\sum_{p=1}^P M_p$ . Thus, the term-question matrix becomes  $\mathbf{D} = \{\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_P\}$  and  $\mathbf{D} \in \mathbb{R}^{(\sum_{p=1}^P M_p) \times N}$ . However, there are two problems with this enrichment: (1) enriching the original questions with the translated words from other languages makes the question representation even more sparse; (2) statistical machine translation may introduce noise.<sup>5</sup> To solve these two problems, we propose to leverage statistical machine translation to improve question retrieval via matrix factorization. Figure 1 presents the framework of our proposed method, where  $q_i$  represents a queried question, and  $\mathbf{q}_i$  is a vector representation of  $q_i$ .

<sup>5</sup>Statistical machine translation quality is far from satisfactory in real applications.

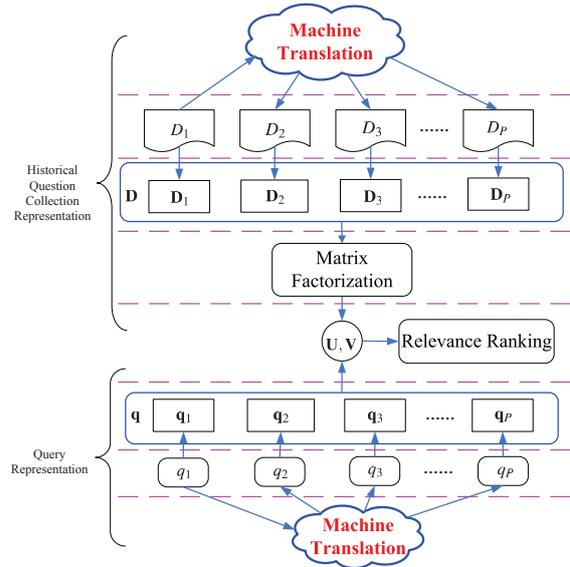


Figure 1: Framework of our proposed approach for question retrieval.

### 2.2 Model Formulation

To tackle the data sparseness of question representation with the translated words, we hope to find two or more lower dimensional matrices whose product provides a good approximate to the original one via matrix factorization. Previous studies have shown that there is psychological and physiological evidence for parts-based representation in the human brain (Wachsmuth et al., 1994). The non-negative matrix factorization (NMF) is proposed to learn the parts of objects like text documents (Lee and Seung, 2001). NMF aims to find two non-negative matrices whose product provides a good approximation to the original matrix and has been shown to be superior to SVD in document clustering (Xu et al., 2003; Tang et al., 2012).

In this paper, NMF is used to induce the reduced representation  $\mathbf{V}_p$  of  $\mathbf{D}_p$ ,  $\mathbf{D}_p$  is independent on  $\{\mathbf{D}_1, \mathbf{D}_2, \dots, \mathbf{D}_{p-1}, \mathbf{D}_{p+1}, \dots, \mathbf{D}_P\}$ . When ignoring the coupling between  $\mathbf{V}_p$ , it can be solved by minimizing the objective function as follows:

$$\mathcal{O}_1(\mathbf{U}_p, \mathbf{V}_p) = \min_{\mathbf{U}_p \geq 0, \mathbf{V}_p \geq 0} \|\mathbf{D}_p - \mathbf{U}_p \mathbf{V}_p\|_F^2 \quad (1)$$

where  $\|\cdot\|_F$  denotes Frobenius norm of a matrix. Matrices  $\mathbf{U}_p \in \mathbb{R}^{M_p \times K}$  and  $\mathbf{V}_p \in \mathbb{R}^{K \times N}$  are the reduced representation for terms and questions in the  $K$  dimensional space, respectively.

To reduce the noise introduced by statistical machine translation, we assume that  $\mathbf{V}_p$  from language  $\mathbf{D}_p$  ( $p \in [2, P]$ ) should be close to  $\mathbf{V}_1$

from the original language  $\mathbf{D}_1$ . Based on this assumption, we minimize the distance between  $\mathbf{V}_p$  ( $p \in [2, P]$ ) and  $\mathbf{V}_1$  as follows:

$$\mathcal{O}_2(\mathbf{V}_p) = \min_{\mathbf{V}_p \geq 0} \sum_{p=2}^P \|\mathbf{V}_p - \mathbf{V}_1\|_F^2 \quad (2)$$

Combining equations (1) and (2), we get the following objective function:

$$\begin{aligned} \mathcal{O}(\mathbf{U}_1, \dots, \mathbf{U}_P; \mathbf{V}_1, \dots, \mathbf{V}_P) \\ = \sum_{p=1}^P \|\mathbf{D}_p - \mathbf{U}_p \mathbf{V}_p\|_F^2 + \sum_{p=2}^P \lambda_p \|\mathbf{V}_p - \mathbf{V}_1\|_F^2 \end{aligned} \quad (3)$$

where parameter  $\lambda_p$  ( $p \in [2, P]$ ) is used to adjust the relative importance of these two components. If we set a small value for  $\lambda_p$ , the objective function behaves like the traditional NMF and the importance of data sparseness is emphasized; while a big value of  $\lambda_p$  indicates  $\mathbf{V}_p$  should be very closed to  $\mathbf{V}_1$ , and equation (3) aims to remove the noise introduced by statistical machine translation.

By solving the optimization problem in equation (4), we can get the reduced representation of terms and questions.

$$\begin{aligned} \min \mathcal{O}(\mathbf{U}_1, \dots, \mathbf{U}_P; \mathbf{V}_1, \dots, \mathbf{V}_P) \\ \text{subject to : } \mathbf{U}_p \geq 0, \mathbf{V}_p \geq 0, p \in [1, P] \end{aligned} \quad (4)$$

### 2.3 Optimization

The objective function  $\mathcal{O}$  defined in equation (4) performs data sparseness and noise removing simultaneously. There are  $2P$  coupling components in  $\mathcal{O}$ , and  $\mathcal{O}$  is not convex in both  $\mathbf{U}$  and  $\mathbf{V}$  together. Therefore it is unrealistic to expect an algorithm to find the global minima. In the following, we introduce an iterative algorithm which can achieve local minima. In our optimization framework, we optimize the objective function in equation (4) by alternatively minimizing each component when the remaining  $2P - 1$  components are fixed. This procedure is summarized in Algorithm 1.

#### 2.3.1 Update of Matrix $\mathbf{U}_p$

Holding  $\mathbf{V}_1, \dots, \mathbf{V}_P$  and  $\mathbf{U}_1, \dots, \mathbf{U}_{p-1}, \mathbf{U}_{p+1}, \dots, \mathbf{U}_P$  fixed, the update of  $\mathbf{U}_p$  amounts to the following optimization problem:

$$\min_{\mathbf{U}_p \geq 0} \|\mathbf{D}_p - \mathbf{U}_p \mathbf{V}_p\|_F^2 \quad (5)$$

---

#### Algorithm 1 Optimization framework

---

**Input:**  $\mathbf{D}_p \in \mathbb{R}^{m_p \times N}$ ,  $p \in [1, P]$   
1: **for**  $p = 1 : P$  **do**  
2:  $\mathbf{V}_p^{(0)} \in \mathbb{R}^{K \times N} \leftarrow$  random matrix  
3: **for**  $t = 1 : T$  **do**  $\triangleright T$  is iteration times  
4:  $\mathbf{U}_p^{(t)} \leftarrow$  Update $\mathbf{U}(\mathbf{D}_p, \mathbf{V}_p^{(t-1)})$   
5:  $\mathbf{V}_p^{(t)} \leftarrow$  Update $\mathbf{V}(\mathbf{D}_p, \mathbf{U}_p^{(t)})$   
6: **end for**  
7: **return**  $\mathbf{U}_p^{(T)}, \mathbf{V}_p^{(T)}$   
8: **end for**

---



---

#### Algorithm 2 Update $\mathbf{U}_p$

---

**Input:**  $\mathbf{D}_p \in \mathbb{R}^{M_p \times N}$ ,  $\mathbf{V}_p \in \mathbb{R}^{K \times N}$   
1: **for**  $i = 1 : M_p$  **do**  
2:  $\bar{\mathbf{u}}_i^{(p)*} = (\mathbf{V}_p \mathbf{V}_p^T)^{-1} \mathbf{V}_p \bar{\mathbf{d}}_i^{(p)}$   
3: **end for**  
4: **return**  $\mathbf{U}_p$

---

Let  $\bar{\mathbf{d}}_i^{(p)} = (d_{i1}^{(p)}, \dots, d_{iK}^{(p)})^T$  and  $\bar{\mathbf{u}}_i^{(p)} = (u_{i1}^{(p)}, \dots, u_{iK}^{(p)})^T$  be the column vectors whose entries are those of the  $i^{th}$  row of  $\mathbf{D}_p$  and  $\mathbf{U}_p$  respectively. Thus, the optimization of equation (5) can be decomposed into  $M_p$  optimization problems that can be solved independently, with each corresponding to one row of  $\mathbf{U}_p$ :

$$\min_{\bar{\mathbf{u}}_i^{(p)} \geq 0} \|\bar{\mathbf{d}}_i^{(p)} - \mathbf{V}_p^T \bar{\mathbf{u}}_i^{(p)}\|_2^2 \quad (6)$$

for  $i = 1, \dots, M_p$ .

Equation (6) is a standard least squares problems in statistics and the solution is:

$$\bar{\mathbf{u}}_i^{(p)*} = (\mathbf{V}_p \mathbf{V}_p^T)^{-1} \mathbf{V}_p \bar{\mathbf{d}}_i^{(p)} \quad (7)$$

Algorithm 2 shows the procedure.

#### 2.3.2 Update of Matrix $\mathbf{V}_p$

Holding  $\mathbf{U}_1, \dots, \mathbf{U}_P$  and  $\mathbf{V}_1, \dots, \mathbf{V}_{p-1}, \mathbf{V}_{p+1}, \dots, \mathbf{V}_P$  fixed, the update of  $\mathbf{V}_p$  amounts to the optimization problem divided into two categories.

if  $p \in [2, P]$ , the objective function can be written as:

$$\min_{\mathbf{V}_p \geq 0} \|\mathbf{D}_p - \mathbf{U}_p \mathbf{V}_p\|_F^2 + \lambda_p \|\mathbf{V}_p - \mathbf{V}_1\|_F^2 \quad (8)$$

if  $p = 1$ , the objective function can be written as:

$$\min_{\mathbf{V}_p \geq 0} \|\mathbf{D}_p - \mathbf{U}_p \mathbf{V}_p\|_F^2 + \lambda_p \|\mathbf{V}_p\|_F^2 \quad (9)$$

Let  $\mathbf{d}_j^{(p)}$  be the  $j^{\text{th}}$  column vector of  $\mathbf{D}_p$ , and  $\mathbf{v}_j^{(p)}$  be the  $j^{\text{th}}$  column vector of  $\mathbf{V}_p$ , respectively. Thus, equation (8) can be rewritten as:

$$\min_{\{\mathbf{v}_j^{(p)} \geq 0\}} \sum_{j=1}^N \|\mathbf{d}_j^{(p)} - \mathbf{U}_p \mathbf{v}_j^{(p)}\|_2^2 + \sum_{j=1}^N \lambda_p \|\mathbf{v}_j^{(p)} - \mathbf{v}_j^{(1)}\|_2^2 \quad (10)$$

which can be decomposed into  $N$  optimization problems that can be solved independently, with each corresponding to one column of  $\mathbf{V}_p$ :

$$\min_{\mathbf{v}_j^{(p)} \geq 0} \|\mathbf{d}_j^{(p)} - \mathbf{U}_p \mathbf{v}_j^{(p)}\|_2^2 + \lambda_p \|\mathbf{v}_j^{(p)} - \mathbf{v}_j^{(1)}\|_2^2 \quad (11)$$

for  $j = 1, \dots, N$ .

Equation (12) is a least square problem with  $L_2$  norm regularization. Now we rewrite the objective function in equation (12) as

$$\mathcal{L}(\mathbf{v}_j^{(p)}) = \|\mathbf{d}_j^{(p)} - \mathbf{U}_p \mathbf{v}_j^{(p)}\|_2^2 + \lambda_p \|\mathbf{v}_j^{(p)} - \mathbf{v}_j^{(1)}\|_2^2 \quad (12)$$

where  $\mathcal{L}(\mathbf{v}_j^{(p)})$  is convex, and hence has a unique solution. Taking derivatives, we obtain:

$$\frac{\partial \mathcal{L}(\mathbf{v}_j^{(p)})}{\partial \mathbf{v}_j^{(p)}} = -2\mathbf{U}_p^T (\mathbf{d}_j^{(p)} - \mathbf{U}_p \mathbf{v}_j^{(p)}) + 2\lambda_p (\mathbf{v}_j^{(p)} - \mathbf{v}_j^{(1)}) \quad (13)$$

Forcing the partial derivative to be zero leads to

$$\mathbf{v}_j^{(p)*} = (\mathbf{U}_p^T \mathbf{U}_p + \lambda_p \mathbf{I})^{-1} (\mathbf{U}_p^T \mathbf{d}_j^{(p)} + \lambda_p \mathbf{v}_j^{(1)}) \quad (14)$$

where  $p \in [2, P]$  denotes the foreign language representation.

Similarly, the solution of equation (9) is:

$$\mathbf{v}_j^{(p)*} = (\mathbf{U}_p^T \mathbf{U}_p + \lambda_p \mathbf{I})^{-1} \mathbf{U}_p^T \mathbf{d}_j^{(p)} \quad (15)$$

where  $p = 1$  denotes the original language representation.

Algorithm 3 shows the procedure.

## 2.4 Time Complexity Analysis

In this subsection, we discuss the time complexity of our proposed method. The optimization  $\bar{\mathbf{u}}_i^{(p)}$  using Algorithm 2 should calculate  $\mathbf{V}_p \mathbf{V}_p^T$  and  $\mathbf{V}_p \bar{\mathbf{d}}_i^{(p)}$ , which takes  $O(NK^2 + NK)$  operations. Therefore, the optimization  $\mathbf{U}_p$  takes  $O(NK^2 + M_p NK)$  operations. Similarly, the time complexity of optimization  $\mathbf{V}_i$  using Algorithm 3 is  $O(M_p K^2 + M_p NK)$ .

Another time complexity is the iteration times  $T$  used in Algorithm 1 and the total number of

---

## Algorithm 3 Update $\mathbf{V}_p$

---

**Input:**  $\mathbf{D}_p \in \mathbb{R}^{M_p \times N}$ ,  $\mathbf{U}_p \in \mathbb{R}^{M_p \times K}$   
1:  $\Sigma \leftarrow (\mathbf{U}_p^T \mathbf{U}_p + \lambda_p \mathbf{I})^{-1}$   
2:  $\Phi \leftarrow \mathbf{U}_p^T \mathbf{D}_p$   
3: **if**  $p = 1$  **then**  
4:   **for**  $j = 1 : N$  **do**  
5:      $\mathbf{v}_j^{(p)} \leftarrow \Sigma \phi_j$ ,  $\phi_j$  is the  $j^{\text{th}}$  column of  $\Phi$   
6:   **end for**  
7: **end if**  
8: **return**  $\mathbf{V}_1$   
9: **if**  $p \in [2, P]$  **then**  
10:   **for**  $j = 1 : N$  **do**  
11:      $\mathbf{v}_j^{(p)} \leftarrow \Sigma (\phi_j + \lambda_p \mathbf{v}_j^{(1)})$   
12:   **end for**  
13: **end if**  
14: **return**  $\mathbf{V}_p$

---

languages  $P$ , the overall time complexity of our proposed method is:

$$\sum_{p=1}^P T \times O(NK^2 + M_p K^2 + 2M_p NK) \quad (16)$$

For each language  $\mathbf{D}_p$ , the size of vocabulary  $M_p$  is almost constant as the number of questions increases. Besides,  $K \ll \min(M_p, N)$ , theoretically, the computational time is almost linear with the number of questions  $N$  and the number of languages  $P$  considered in the paper. Thus, the proposed method can be easily adapted to the large-scale information retrieval task.

## 2.5 Relevance Ranking

The advantage of incorporating statistical machine translation in relevance ranking is to reduce ‘‘word ambiguity’’ and ‘‘word mismatch’’ problems. To do so, given a queried question  $q$  and a historical question  $d$  from Yahoo! Answers, we first translate  $q$  and  $d$  into other foreign languages (e.g., Chinese, French etc.) and get the corresponding translated representation  $q_i$  and  $d_i$  ( $i \in [2, P]$ ), where  $P$  is the number of languages considered in the paper. For queried question  $q = q_1$ , we represent it in the reduced space:

$$\mathbf{v}_{q_1} = \arg \min_{\mathbf{v} \geq 0} \|\mathbf{q}_1 - \mathbf{U}_1 \mathbf{v}\|_2^2 + \lambda_1 \|\mathbf{v}\|_2^2 \quad (17)$$

where vector  $\mathbf{q}_1$  is the tf-idf representation of queried question  $q_1$  in the term space. Similarly, for historical question  $d = d_1$  (and its tf-idf representation  $\mathbf{d}_1$  in the term space) we represent it in the reduced space as  $\mathbf{v}_{d_1}$ .

The relevance score between the queried question  $q_1$  and the historical question  $d_1$  in the reduced space is, then, calculated as the cosine similarity between  $\mathbf{v}_{q_1}$  and  $\mathbf{v}_{d_1}$ :

$$s(q_1, d_1) = \frac{\langle \mathbf{v}_{q_1}, \mathbf{v}_{d_1} \rangle}{\|\mathbf{v}_{q_1}\|_2 \cdot \|\mathbf{v}_{d_1}\|_2} \quad (18)$$

For translated representation  $q_i$  ( $i \in [2, P]$ ), we also represent it in the reduced space:

$$\mathbf{v}_{q_i} = \arg \min_{\mathbf{v} \geq 0} \|\mathbf{q}_i - \mathbf{U}_i \mathbf{v}\|_2^2 + \lambda_i \|\mathbf{v} - \mathbf{v}_{q_1}\|_2^2 \quad (19)$$

where vector  $\mathbf{q}_i$  is the tf-idf representation of  $q_i$  in the term space. Similarly, for translated representation  $d_i$  (and its tf-idf representation  $\mathbf{d}_i$  in the term space) we also represent it in the reduced space as  $\mathbf{v}_{d_i}$ . The relevance score  $s(q_i, d_i)$  between  $q_i$  and  $d_i$  in the reduced space can be calculated as the cosine similarity between  $\mathbf{v}_{q_i}$  and  $\mathbf{v}_{d_i}$ .

Finally, we consider learning a relevance function of the following general, linear form:

$$Score(q, d) = \boldsymbol{\theta}^T \cdot \boldsymbol{\Phi}(q, d) \quad (20)$$

where feature vector  $\boldsymbol{\Phi}(q, d) = (s_{VSM}(q, d), s(q_1, d_1), s(q_2, d_2), \dots, s(q_P, d_P))$ , and  $\boldsymbol{\theta}$  is the corresponding weight vector, we optimize this parameter for our evaluation metrics directly using the Powell Search algorithm (Paul et al., 1992) via cross-validation.  $s_{VSM}(q, d)$  is the relevance score in the term space and can be calculated using Vector Space Model (VSM).

## 3 Experiments

### 3.1 Data Set and Evaluation Metrics

We collect the data set from Yahoo! Answers and use the *getByCategory* function provided in Yahoo! Answers API<sup>6</sup> to obtain CQA threads from the Yahoo! site. More specifically, we utilize the *resolved* questions and the resulting question repository that we use for question retrieval contains 2,288,607 questions. Each resolved question consists of four parts: “question title”, “question description”, “question answers” and “question category”. For question retrieval, we only use the “question title” part. It is assumed that question title already provides enough semantic information for understanding the users’ information needs (Duan et al., 2008). There are 26 categories

Category	#Size	Category	#Size
Arts & Humanities	86,744	Home & Garden	35,029
Business & Finance	105,453	Beauty & Style	37,350
Cars & Transportation	145,515	Pet	54,158
Education & Reference	80,782	Travel	305,283
Entertainment & Music	152,769	Health	132,716
Family & Relationships	34,743	Sports	214,317
Politics & Government	59,787	Social Science	46,415
Pregnancy & Parenting	43,103	Ding out	46,933
Science & Mathematics	89,856	Food & Drink	45,055
Computers & Internet	90,546	News & Events	20,300
Games & Recreation	53,458	Environment	21,276
Consumer Electronics	90,553	Local Businesses	51,551
Society & Culture	94,470	Yahoo! Products	150,445

Table 2: Number of questions in each first-level category.

at the first level and 1,262 categories at the leaf level. Each question belongs to a unique leaf category. Table 2 shows the distribution across first-level categories of the questions in the archives.

We use the same test set in previous work (Cao et al., 2009; Cao et al., 2010). This set contains 252 queried questions and can be freely downloaded for research communities.<sup>7</sup>

The original language of the above data set is English ( $l_1$ ) and then they are translated into four other languages (Chinese ( $l_2$ ), French ( $l_3$ ), German ( $l_4$ ), Italian ( $l_5$ )), thus the number of language considered is  $P = 5$  by using the state-of-the-art translation tool -- Google Translate.

**Evaluation Metrics:** We evaluate the performance of question retrieval using the following metrics: Mean Average Precision (MAP) and Precision@N (P@N). MAP rewards methods that return relevant questions early and also rewards correct ranking of the results. P@N reports the fraction of the top- $N$  questions retrieved that are relevant. We perform a significant test, i.e., a  $t$ -test with a default significant level of 0.05.

We tune the parameters on a small development set of 50 questions. This development set is also extracted from Yahoo! Answers, and it is not included in the test set. For parameter  $K$ , we do an experiment on the development set to determine the optimal values among 50, 100, 150,  $\dots$ , 300 in terms of MAP. Finally, we set  $K = 100$  in the experiments empirically as this setting yields the best performance. For parameter  $\lambda_1$ , we set  $\lambda_1 = 1$  empirically, while for parameter  $\lambda_i$  ( $i \in [2, P]$ ), we set  $\lambda_i = 0.25$  empirically and ensure that  $\sum_i \lambda_i = 1$ .

<sup>6</sup><http://developer.yahoo.com/answers>

<sup>7</sup><http://homepages.inf.ed.ac.uk/gcong/qa/>

#	Methods	MAP	P@10
1	VSM	0.242	0.226
2	LM	0.385	0.242
3	Jeon et al. (2005)	0.405	0.247
4	Xue et al. (2008)	0.436	0.261
5	Zhou et al. (2011)	0.452	0.268
6	Singh (2012)	0.450	0.267
7	Zhou et al. (2012)	0.483	0.275
8	SMT + MF ( $P = 2, l_1, l_2$ )	0.527	0.284
9	<b>SMT + MF (<math>P = 5</math>)</b>	<b>0.564</b>	<b>0.291</b>

Table 3: Comparison with different methods for question retrieval.

### 3.2 Question Retrieval Results

Table 3 presents the main retrieval performance. Row 1 and row 2 are two baseline systems, which model the relevance score using VSM (Cao et al., 2010) and language model (LM) (Zhai and Lafferty, 2001; Cao et al., 2010) in the term space. Row 3 and row 6 are monolingual translation models to address the word mismatch problem and obtain the state-of-the-art performance in previous work. Row 3 is the word-based translation model (Jeon et al., 2005), and row 4 is the word-based translation language model, which linearly combines the word-based translation model and language model into a unified framework (Xue et al., 2008). Row 5 is the phrase-based translation model, which translates a sequence of words as whole (Zhou et al., 2011). Row 6 is the entity-based translation model, which extends the word-based translation model and explores strategies to learn the translation probabilities between words and the concepts using the CQA archives and a popular entity catalog (Singh, 2012). Row 7 is the bilingual translation model, which translates the English questions from Yahoo! Answers into Chinese questions using Google Translate and expands the English words with the translated Chinese words (Zhou et al., 2012). For these previous work, we use the same parameter settings in the original papers. Row 8 and row 9 are our proposed method, which leverages statistical machine translation to improve question retrieval via matrix factorization. In row 8, we only consider two languages (English and Chinese) and translate English questions into Chinese using Google Translate in order to compare with Zhou et al. (2012). In row 9, we translate English questions into other four languages. There are some clear trends in the result of Table 3:

(1) Monolingual translation models significantly outperform the VSM and LM (row 1 and

row 2 vs. row 3, row 4, row 5 and row 6).

(2) Taking advantage of potentially rich semantic information drawn from other languages via statistical machine translation, question retrieval performance can be significantly improved (row 3, row 4, row 5 and row 6 vs. row 7, row 8 and row 9, all these comparisons are statistically significant at  $p < 0.05$ ).

(3) Our proposed method (leveraging statistical machine translation via matrix factorization, SMT + MF) significantly outperforms the bilingual translation model of Zhou et al. (2012) (row 7 vs. row 8, the comparison is statistically significant at  $p < 0.05$ ). The reason is that matrix factorization used in the paper can effectively solve the data sparseness and noise introduced by the machine translator simultaneously.

(4) When considering more languages, question retrieval performance can be further improved (row 8 vs. row 9).

Note that Wang et al. (2009) also addressed the word mismatch problem for question retrieval by using syntactic tree matching. We do not compare with Wang et al. (2009) in Table 3 because previous work (Ming et al., 2010) demonstrated that word-based translation language model (Xue et al., 2008) obtained the superior performance than the syntactic tree matching (Wang et al., 2009). Besides, some other studies attempt to improve question retrieval with category information (Cao et al., 2009; Cao et al., 2010), label ranking (Li et al., 2011) or world knowledge (Zhou et al., 2012). However, their methods are orthogonal to ours, and we suspect that combining the category information or label ranking into our proposed method might get even better performance. We leave it for future research.

### 3.3 Impact of the Matrix Factorization

Our proposed method (SMT + MF) can effectively solve the data sparseness and noise via matrix factorization. To further investigate the impact of the matrix factorization, one intuitive way is to expand the original questions with the translated words from other four languages, without considering the data sparseness and noise introduced by machine translator. We compare our SMT + MF with this intuitive enriching method (SMT + IEM). Besides, we also employ our proposed matrix factorization to the original question representation (VSM + MF). Table 4 shows the comparison.

#	Methods	MAP	P@10
1	VSM	0.242	0.226
2	VSM + MF	0.411	0.253
3	SMT + IEM ( $P = 5$ )	0.495	0.280
4	SMT + MF ( $P = 5$ )	0.564	0.291

Table 4: The impact of matrix factorization.

(1) Our proposed matrix factorization can significantly improve the performance of question retrieval (row 1 vs. row 2; row 3 vs. row 4, the improvements are statistically significant at  $p < 0.05$ ). The results indicate that our proposed matrix factorization can effectively address the issues of data sparseness and noise introduced by statistical machine translation.

(2) Compared to the relative improvements of row 3 and row 4, the relative improvements of row 1 and row 2 is much larger. The reason may be that although matrix factorization can be used to reduce dimension, it may impair the meaningful terms.

(3) Compared to VSM, the performance of SMT + IEM is significantly improved (row 1 vs. row 3), which supports the motivation that the word ambiguity and word mismatch problems could be partially addressed by Google Translate.

### 3.4 Impact of the Translation Language

One of the success of this paper is to take advantage of potentially rich semantic information drawn from other languages to solve the word ambiguity and word mismatch problems. So we construct a dummy translator (DT) that translates an English word to itself. Thus, through this translation, we do not add any semantic information into the original questions. The comparison is presented in Table 5. Row 1 (DT + MF) represents integrating two copies of English questions with our proposed matrix factorization. From Table 5, we have several different findings:

(1) Taking advantage of potentially rich semantic information drawn from other languages can significantly improve the performance of question retrieval (row 1 vs. row 2, row 3, row 4 and row 5, the improvements relative to DT + MF are statistically significant at  $p < 0.05$ ).

(2) Different languages contribute unevenly for question retrieval (e.g., row 2 vs. row 3). The reason may be that the improvements of leveraging different other languages depend on the quality of machine translation. For example, row 3

#	Methods	MAP
1	DT + MF ( $l_1, l_1$ )	0.352
2	SMT + MF ( $P = 2, l_1, l_2$ )	0.527
3	SMT + MF ( $P = 2, l_1, l_3$ )	0.553
4	SMT + MF ( $P = 2, l_1, l_4$ )	0.536
5	SMT + MF ( $P = 2, l_1, l_5$ )	0.545
6	SMT + MF ( $P = 3, l_1, l_2, l_3$ )	0.559
7	SMT + MF ( $P = 4, l_1, l_2, l_3, l_4$ )	0.563
8	SMT + MF ( $P = 5, l_1, l_2, l_3, l_4, l_5$ )	0.564

Table 5: The impact of translation language.

Method	Translation	MAP
SMT + MF ( $P = 2, l_1, l_2$ )	Dict	0.468
	GTrans	0.527

Table 6: Impact of the contextual information.

is better than row 2 because the translation quality of English-French is much better than English-Chinese.

(3) Using much more languages does not seem to produce significantly better performance (row 6 and row 7 vs. row 8). The reason may be that inconsistency between different languages may exist due to statistical machine translation.

### 3.5 Impact of the Contextual Information

In this paper, we translate the English questions into other four languages using Google Translate (GTrans), which takes into account contextual information during translation. If we translate a question word by word, it discards the contextual information. We would expect that such a translation would not be able to solve the word ambiguity problem.

To investigate the impact of contextual information for question retrieval, we only consider two languages and translate English questions into Chinese using an English to Chinese lexicon (Dict) in StarDict<sup>8</sup>. Table 6 shows the experimental results, we can see that the performance is degraded when the contextual information is not considered for the translation of questions. The reason is that GTrans is context-dependent and thus produces different translated Chinese words depending on the context of an English word. Therefore, the word ambiguity problem can be solved during the English-Chinese translation.

## 4 Conclusions and Future Work

In this paper, we propose to employ statistical machine translation to improve question retrieval and

<sup>8</sup>StarDict is an open source dictionary software, available at <http://stardict.sourceforge.net/>.

enrich the question representation with the translated words from other languages via matrix factorization. Experiments conducted on a real CQA data show some promising findings: (1) the proposed method significantly outperforms the previous work for question retrieval; (2) the proposed matrix factorization can significantly improve the performance of question retrieval, no matter whether considering the translation languages or not; (3) considering more languages can further improve the performance but it does not seem to produce significantly better performance; (4) different languages contribute unevenly for question retrieval; (5) our proposed method can be easily adapted to the large-scale information retrieval task.

As future work, we plan to incorporate the question structure (e.g., question topic and question focus (Duan et al., 2008)) into the question representation for question retrieval. We also want to further investigate the use of the proposed method for other kinds of data set, such as categorized questions from forum sites and FAQ sites.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61070106, No. 61272332 and No. 61202329), the National High Technology Development 863 Program of China (No. 2012AA011102), the National Basic Research Program of China (No. 2012CB316300), We thank the anonymous reviewers for their insightful comments. We also thank Dr. Gao Cong for providing the data set and Dr. Li Cai for some discussion.

## References

- L. Adamic, J. Zhang, E. Bakshy, and M. Ackerman. 2008. Knowledge sharing and yahoo answers: everyone knows and something. In *Proceedings of WWW*.
- A. Berger, R. Caruana, D. Cohn, D. Freitag, and V. Mittal. 2000. Bridging the lexical chasm: statistical approach to answer-finding. In *Proceedings of SIGIR*, pages 192-199.
- D. Bernhard and I. Gurevych. 2009. Combining lexical semantic resources with question & answer archives for translation-based answer finding. In *Proceedings of ACL*, pages 728-736.
- P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263-311.
- X. Cao, G. Cong, B. Cui, C. Jensen, and C. Zhang. 2009. The use of categorization information in language models for question retrieval. In *Proceedings of CIKM*, pages 265-274.
- X. Cao, G. Cong, B. Cui, and C. Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *Proceedings of WWW*, pages 201-210.
- H. Duan, Y. Cao, C. Y. Lin, and Y. Yu. 2008. Searching questions by identifying questions topics and question focus. In *Proceedings of ACL*, pages 156-164.
- C. L. Lawson and R. J. Hanson. 1974. Solving least squares problems. *Prentice-Hall*.
- J. -T. Lee, S. -B. Kim, Y. -I. Song, and H. -C. Rim. 2008. Bridging lexical gaps between queries and questions on large online Q&A collections with compact translation models. In *Proceedings of EMNLP*, pages 410-418.
- W. Wang, B. Li, and I. King. 2011. Improving question retrieval in community question answering with label ranking. In *Proceedings of IJCNN*, pages 349-356.
- D. D. Lee and H. S. Seung. 2001. Algorithms for non-negative matrix factorization. In *Proceedings of NIPS*.
- Z. Ming, K. Wang, and T. -S. Chua. 2010. Prototype hierarchy based clustering for the categorization and navigation of web collections. In *Proceedings of SIGIR*, pages 2-9.
- J. Jeon, W. Croft, and J. Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of CIKM*, pages 84-90.
- C. Paige and M. Saunders. 1982. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Transaction on Mathematical Software*, 8(1):43-71.
- W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 1992. *Numerical Recipes In C*. Cambridge Univ. Press.
- S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, pages 464-471.
- A. Singh. 2012. Entity based q&a retrieval. In *Proceedings of EMNLP-CoNLL*, pages 1266-1277.
- J. Tang, X. Wang, H. Gao, X. Hu, and H. Liu. 2012. Enriching short text representation in microblog for clustering. *Front. Comput.*, 6(1):88-101.

- E. Wachsmuth, M. W. Oram, and D. I. Perrett. 1994. Recognition of objects and their component parts: responses of single units in the temporal cortex of teh macaque. *Cerebral Cortex*, 4:509-522.
- K. Wang, Z. Ming, and T-S. Chua. 2009. A syntactic tree matching approach to find similar questions in community-based qa services. In *Proceedings of SIGIR*, pages 187-194.
- B. Wang, X. Wang, C. Sun, B. Liu, and L. Sun. 2010. Modeling semantic relevance for question-answer pairs in web social communities. In *Proceedings of ACL*, pages 1230-1238.
- W. Xu, X. Liu, and Y. Gong. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of SIGIR*, pages 267-273.
- X. Xue, J. Jeon, and W. B. Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of SIGIR*, pages 475-482.
- C. Zhai and J. Lafferty. 2001. A study of smooth methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR*, pages 334-342.
- G. Zhou, L. Cai, J. Zhao, and K. Liu. 2011. Phrase-based translation model for question retrieval in community question answer archives. In *Proceedings of ACL*, pages 653-662.
- G. Zhou, K. Liu, and J. Zhao. 2012. Exploiting bilingual translation for question retrieval in community-based question answering. In *Proceedings of COLING*, pages 3153-3170.
- G. Zhou, Y. Liu, F. Liu, D. Zeng, and J. Zhao. 2013. Improving Question Retrieval in Community Question Answering Using World Knowledge. In *Proceedings of IJCAI*.

# Improved Lexical Acquisition through DPP-based Verb Clustering

**Roi Reichart**

University of Cambridge, UK  
rr439@cam.ac.uk

**Anna Korhonen**

University of Cambridge, UK  
alk23@cam.ac.uk

## Abstract

Subcategorization frames (SCFs), selectional preferences (SPs) and verb classes capture related aspects of the predicate-argument structure. We present the first unified framework for unsupervised learning of these three types of information. We show how to utilize Determinantal Point Processes (DPPs), elegant probabilistic models that are defined over the possible subsets of a given dataset and give higher probability mass to high quality and diverse subsets, for clustering. Our novel clustering algorithm constructs a joint SCF-DPP DPP kernel matrix and utilizes the efficient sampling algorithms of DPPs to cluster together verbs with similar SCFs and SPs. We evaluate the induced clusters in the context of the three tasks and show results that are superior to strong baselines for each <sup>1</sup>.

## 1 Introduction

Verb classes (VCs), subcategorization frames (SCFs) and selectional preferences (SPs) capture different aspects of predicate-argument structure. SCFs describe the syntactic realization of verbal predicate-argument structure, SPs capture the semantic preferences verbs have for their arguments and VCs in the Levin (1993) tradition provide a shared level of abstraction for verbs that share many aspects of their syntactic and semantic behavior.

These three of types of information have proved useful for Natural Language Processing (NLP)

<sup>1</sup>The source code of the clustering algorithms and evaluation is submitted with this paper and will be made publicly available upon acceptance of the paper.

tasks which require information about predicate-argument structure, including parsing (Shi and Mihalcea, 2005; Cholakov and van Noord, 2010; Zhou et al., 2011), semantic role labeling (Swier and Stevenson, 2004; Dang, 2004; Bharati et al., 2005; Moschitti and Basili, 2005; zap, 2008; Zapi- rain et al., 2009), and word sense disambiguation (Dang, 2004; Thater et al., 2010; Ó Séaghdha and Korhonen, 2011), among many others.

Because lexical information is highly sensitive to domain variation, approaches that can identify VCs, SCFs and SPs in corpora have become increasingly popular, e.g. (O'Donovan et al., 2005; Schulte im Walde, 2006; Erk, 2007; Preiss et al., 2007; Van de Cruys, 2009; Reisinger and Mooney, 2011; Sun and Korhonen, 2011; Lippincott et al., 2012).

The task of SCF induction involves identifying the arguments of a verb lemma and generalizing about the frames (i.e. SCFs) taken by the verb, where each frame includes a number of arguments and their syntactic types. For example, in (1), the verb "show" takes the frame SUBJ-DOBJ-CCOMP (subject, direct object, and clausal complement).

(1) [A number of SCF acquisition papers]SUBJ [show]VERB [their readers]DOBJ [which features are most valuable for the acquisition process]CCOMP.

SP induction involves identifying and classifying the lexical items in a given argument slot. In sentence (2), for example, the verb "show" takes the frame SUBJ-DOBJ. The direct object in this frame is likely to be inanimate.

(2) [Most SCF and SP acquisition papers]SUBJ,

[show]VERB [no evidence to the usefulness of joint learning leaning for these tasks]DOBJ.

Finally, VC induction involves clustering together verbs with similar meaning, reflected in similar SCFs and SPs. For example, "show" in the above examples could get clustered together with "demonstrate" and "indicate".

Because these challenging tasks capture complementary information about predicate argument structure, they should be able to inform and support each other. Recently, researchers have begun to investigate the benefits of their joint learning. Schulte im Walde et al. (2008) integrated SCF and VC acquisition and used it for WordNet-based SP classification. Ó Séaghdha (2010) presented a "dual-topic" model for SPs that induces also verb clusters. Both works reported SP evaluation with promising results. Lippincott et al. (2012) presented a joint model for inducing simple syntactic frames and VCs. They reported high accuracy results on VCs. de Cruys et al. (2012) introduced a joint model for SCF and SP acquisition. They evaluated both the SCFs and SPs, obtaining reasonable result on both tasks.

In this paper, we present the first unified framework for unsupervised learning of the three types of information - SCFs, SPs and VCs. Our framework is based on Determinantal Point Processes (DPPs, (Kulesza, 2012; Kulesza and Taskar, 2012c)), elegant probabilistic models that are defined over the possible subsets of a given dataset and give higher probability mass to high quality and diverse subsets.

We first show how individual-task DPP kernel matrices can be naturally combined to construct a joint kernel. We use this to construct a joint SCF-SP kernel. We then introduce a novel clustering algorithm based on iterative DPP sampling which can (contrary to other probabilistic frameworks such as Markov random fields) be performed both accurately and efficiently. When defined over the joint SCF and SP kernel, this new algorithm can be used to induce VCs that are valuable for both tasks.

We also contribute by evaluating the value of the clusters induced by our model for the acquisition of the three information types. Our evaluation against a well-known VC gold standard shows that our clustering model outperforms the state-of-the-art verb clustering algorithm of Sun and Korhonen

(2009), in our setup where no manually created SCF or SP data is available. Our evaluation against a well-known SCF gold standard and in the context of SP disambiguation tasks shows results that are superior to strong baselines, demonstrating the benefit our approach.

## 2 Previous Work

**SCF acquisition** Most current works induce SCFs from the output of an unlexicalized parser (i.e. a parser trained without SCF annotations) using hand-written rules (Briscoe and Carroll, 1997; Korhonen, 2002; Preiss et al., 2007) or grammatical relation (GR) co-occurrence statistics (O'Donovan et al., 2005; Chesley and Salmon-Alt, 2006; Ienco et al., 2008; Messiant et al., 2008; Lenci et al., 2008; Altamirano and Alonso i Alemany, 2010; Kawahara and Kurohashi, 2010).

Only a handful of SCF induction works are unsupervised. Carroll and Rooth (1996) applied an EM-based approach to a context-free grammar based model, Dkebowski (2009) used point-wise co-occurrence of arguments in parsed Polish data and Lippincott et al. (2012) presented a Bayesian network model for syntactic frame induction that identifies SPs on argument types. However, the frames induced by Lippincott et al. (2012) do not capture sets of arguments for verbs so are far simpler than traditional SCFs.

Current approaches to SCF acquisition suffer from lack of semantic information which is needed to guide the purely syntax-driven acquisition process. Previous works have showed the benefit of hand-coded semantic information in SCF acquisition (Korhonen, 2002). We will address this problem in an unsupervised way: our approach is to consider SCFs together with semantic SPs through VCs which generalize over syntactically and semantically similar verbs.

**SP acquisition** Considerable research has been conducted on SP acquisition, with a variety of unsupervised models proposed for this task that use no hand-crafted information during training. The latter approaches include latent variable models (Ó Séaghdha, 2010; Ritter and Etzioni, 2010; Reisinger and Mooney, 2011), distributional similarity methods (Bhagat et al., 2007; Basili et al., 2007; Erk, 2007) and methods based on non-negative tensor factorization (Van de Cruys, 2009). These works use a variety of linguistic features in the acquisition process but none of them

integrates the three information types covered in our work.

**Verb clustering** A variety of VC approaches have been proposed in the literature. These include syntactic, semantic and mixed syntactic-semantic classifications (Grishman et al., 1994; Miller, 1995; Baker et al., 1998; Palmer et al., 2005; Schuler, 2006; Hovy et al., 2006). We focus on Levin style classes (Levin, 1993) which are defined in terms of diathesis alternations and capture generalizations over a range of syntactic and semantic properties. Previous unsupervised VC acquisition approaches clustered a variety of linguistic features using different (e.g. K-means and spectral) algorithms (Schulte im Walde, 2006; Joanis et al., 2008; Sun et al., 2008; Li and Brew, 2008; Korhonen et al., 2008; Sun and Korhonen, 2009; Vlachos et al., 2009; Sun and Korhonen, 2011). The linguistic features included SCFs and SPs, but these were induced separately and then feeded as features to the clustering algorithm. Our framework combines together SCF-motivated and SP-motivated kernel matrices, and uses the joint kernel to induce verb clusters which are likely to be highly relevant for both tasks. Importantly, no manual or automatic system for SCF or SP acquisition has been utilized when constructing the kernel matrices, we only consider features extracted from the output of an unlexicalized parser. Our approach hence provides a framework for acquiring valuable information for the three tasks together.

**Joint Modeling** A small number of works have recently investigated joint approaches to SCFs, SPs and VCs. Each of them has addressed only a subset of the tasks and all but one have evaluated the performance in the context of one task only. Ó Séaghdha (2010) presented a “dual-topic” model for SPs that induces VCs, reporting evaluation of SPs only. Lippincott et al. (2012) presented a Bayesian network model for syntactic frame (rather than full SCF) induction that induces VCs. Only VCs are evaluated. de Cruys et al. (2012) presented a joint unsupervised model of SCF and SP acquisition based on non-negative tensor factorization. Both SCFs and SPs were evaluated. Finally, the model of Schulte im Walde et al. (2008) addresses the three types of information but SP parameters are estimated with a WordNet based method and only the SPs are evaluated. Although evaluation of these recent joint models has been partial, the results have been encouraging and fur-

ther motivate the development of a framework that acquires the three types of information together.

### 3 The Unified Framework

In this section we present our unified framework. Our idea is to utilize DPPs for verb clustering that informs both SCF and SP acquisition. DPPs define a probability distribution over the possible subsets of a given set. These models assign higher probability mass to subsets that are both high quality and diverse.

Our novel clustering algorithm makes use of three DPP properties that are appealing for our purpose: (1) The existence of efficient sampling algorithms for these models, which enable tractable sampling of high quality and diverse verb subsets; (2) Such verb subsets form natural high quality seeds for hierarchical clustering; and (3) Given individual-task DPP kernel matrices there are various simple and natural ways to combine them into a new DPP kernel matrix.

Individual task DPP kernels represent (i) the quality of a data point (verb) as its average feature-based similarity with the other points in the data set and (ii) the divergence between a pair of points as the inverse similarity between them. For different tasks, different feature sets are used for the kernel construction. The high quality and diverse subsets sampled from the DPP model are considered good cluster seeds as they are likely to be relatively uniformly spread and to provide good coverage of the data set. The algorithm induces an hierarchical clustering, which is particularly suitable for semantic tasks, where a set of clusters that share a parent consists of pure members (i.e. most of the points in each cluster member belong to the same gold cluster) and together provide good coverage of the verb space.

After a brief description of the Determinantal Point Processes (DPP) framework (Section 3.1), we discuss the construction of the joint DPP kernel, given a kernel for each individual task, In section 3.3 we present the DPP-Cluster clustering algorithm.

#### 3.1 Determinantal Point Processes

Determinantal point processes (DPPs) are elegant probabilistic models of repulsion that offer efficient and exact algorithms for sampling, marginalization, conditioning, and other inference tasks. Recently (Kulesza, 2012; Kulesza and Taskar,

2012c) introduced them to the machine learning community and demonstrated their usefulness for a variety of tasks including document summarization, image search, modeling non-overlapping human poses in images and video and automatically building timelines of important news stories (Kulesza and Taskar, 2010; Kulesza and Taskar, 2012a; Gillenwater et al., 2012; Kulesza and Taskar, 2012b). Below we provide a brief description of the framework, a comprehensive survey can be found in (Kulesza and Taskar, 2012c).

Given a set of items  $\mathcal{Y} = \{y_1, \dots, y_N\}$ , a DPP  $\mathcal{P}$  defines a probability measure on the set of all subsets of  $\mathcal{Y}$ ,  $2^{\mathcal{Y}}$ . Kulesza and Taskar (2012c) restricted their discussion of DPPs to L-ensembles, where the probability of a subset  $\mathbf{Y} \in \mathcal{Y}$  is defined through a positive semi-definite matrix  $L$  indexed by the elements of  $\mathcal{Y}$ :

$$\mathcal{P}_L(\mathbf{Y} = Y) = \frac{\det(L_Y)}{\sum_{Y \subseteq \mathcal{Y}} \det(L_Y)} = \frac{\det(L_Y)}{\det(L + I)} \quad (1)$$

Where  $I$  is the  $N \times N$  identity matrix and  $\det(L_\phi) = 1$ . Since  $L$  is positive semi-definite, it can be decomposed to  $L = B^T B$ . This allows the construction of an intuitively interpretable model where each column  $B_i$  is the product of a quality term  $q_i \in R^+$  and a vector of (normalized) diversity features  $\phi_i \in R^D$ ,  $\|\phi_i\| = 1$ . In this model,  $q_i$  measures an inherent quality of the  $i$ -th item in  $\mathcal{Y}$  while  $\phi_i^T \phi_j \in [-1, 1]$  is a similarity measure between items  $i$  and  $j$ . With this representation we can write:

$$L_{ij} = q_i \phi_i^T \phi_j q_j \quad (2)$$

$$S_{ij} = \phi_i^T \phi_j = \frac{L_{ij}}{\sqrt{L_{ii} L_{jj}}} \quad (3)$$

$$\mathcal{P}_L(\mathbf{Y} = Y) \propto \left( \prod_{i \in Y} q_i^2 \right) \det(S_Y) \quad (4)$$

It can be shown that the first term in equation 4 increases with the quality of the selected items, and the second term increases with their diversity. As a consequence, this distribution places most of its weight on sets that are both high quality and diverse.

Although the number of possible realizations of  $Y$  is exponential in  $N$ , many inference procedures can be performed accurately and efficiently (i.e. in polynomial time which is very short in practice). In particular, sampling, which NP-hard for

alternative models such as Markov Random Fields (MRFs), is efficient, theoretically and practically, for DPPs.

### 3.2 Constructing a Joint Kernel Matrix

DPPs are particularly suitable for joint modeling as they come with various simple and intuitive ways to combine individual model kernel matrices into a joint kernel. This stems from the fact that every positive-semidefinite matrix forms a legal DPP kernel (equation 1). Given individual model DPP kernels, we would therefore like to combine them into a positive-semidefinite matrix.

While there are various ways to construct a positive-semidefinite matrix from two positive-semidefinite matrices – for example, by taking their sum – in this work we are motivated by the product of experts approach (Hinton, 2002), reasoning that high quality assignments according to a product of models have to be of high quality according to each individual model, and sick for a product combination.<sup>2</sup>

In practice we construct the joint kernel in the following way. We build on the aforementioned property that a matrix  $L$  is positive semi-definite iff  $L = B^T B$ . Given two DPPs,  $\mathcal{P}_{L^1}$  defined by  $L^1 = A_1^T A_1$  and  $\mathcal{P}_{L^2}$  defined by  $L^2 = A_2^T A_2$ , we construct the joint kernel  $L^{12}$ :

$$L^{12} = L^1 L^2 L^2 L^1 = C^T C \quad (5)$$

Where  $C = A_2^T A_2 A_1^T A_1$  and  $C^T = A_1^T A_1 A_2^T A_2$ .

### 3.3 Clustering Algorithm

Algorithm (1) and Figure (1) provide a pseudocode of the algorithm and an example output. Below is a detailed description.

**Features** Our algorithm builds two DPP kernel matrices (the *GenKernelMatrix* function), in which the rows and columns correspond to the verbs in the data set, such that the  $(i, j)$ -th entry corresponds to verbs number  $i$  and  $j$ . Following equations 2 and 3 one matrix is built for SCF and one for SP, and they are then combined into the

<sup>2</sup>Note that we do not take a product of the individual models but only of their kernel matrices. Yet, if we construct the joint matrix by a multiplication then it follows from a simple generalization of the Cauchy-Binet formula that its principle minors, which define the subset probabilities (equation 1), are a sum of multiplications of the principle minors of the individual model kernels. Still, we do not have guarantees that our choice of kernel combination is the right one. We leave this for future research.

joint kernel matrix (the *GenJointMat* function) following equation 5. Each kernel matrix requires a proper feature representation  $\phi$  and quality score  $q$ .

In both kernels we represent a verb by the counts of the grammatical relations (GRs) it participates in. In the SCF kernel a GR is represented by the GR type and the POS tags of the verb and its arguments. In the SP kernels the GRs are represented by the POS tags of the verb and its arguments as well as by the argument head word. Based on this feature representation, the similarity (opposite divergence) is encoded to the model by equation 3 as the dot product between the normalized feature vectors. The quality score  $q_i$  of the  $i$ -th verb is the average similarity of this verb with the other verbs in the dataset.

**Cluster set construction** In its while loop, the algorithm iteratively generates fixed-size cluster sets such that each data point belongs to exactly one cluster in one set. These cluster sets form the leaf level of the tree in Figure (1). It does so by extracting the  $T$  highest probability  $K$ -point samples from a set of  $M$  subsets, each of which sampled from the joint DPP model, and clustering them by the *cluster* procedure. The sampling is done by the K-DPP sampling process ((Kulesza and Taskar, 2012c), page 62)<sup>3</sup>.

The *cluster* procedure first seeds a  $K$ -cluster set with the highest probability sample. Then, it gradually extends the clusters by iteratively mapping the samples, in decreasing order of probability, to the existing clusters (the *m1Mapping* function). Mapping is done by attaching every point in the mapped subset to its closet cluster, where the distance between a point and the cluster is the maximum over the distances between the point and each of the points in the cluster. The mapping is many-to-one, that is, multiple points in the subset can be assigned to the same cluster.

Based on the DPP properties, the higher the probability of a sampled subset, the more likely it is to consist of distinct points that provide a good coverage of the verb set. By iteratively extending the clusters with high probability subsets, we thus expect each cluster set to consist of clusters that demonstrate these properties.

<sup>3</sup>K-DPP is a DPP conditioned on the sample size. As shown in ((Kulesza and Taskar, 2012c), Section 2.4.3) this conditional distribution is also a DPP. We could have obtained samples of size  $K$  by sampling the DPP and rejecting samples of other sizes but this would have been slower.

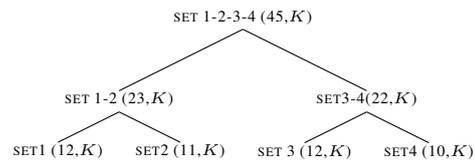


Figure 1: An example output hierarchy of DPP-Cluster for a set of 45 data points. Each set is augmented with the number of points (left number) and clusters (right number) it includes. The iterative DPP-samples clustering (the While loop) generates the lowest level of the tree, by dividing the data set into cluster sets, each of which consists of  $K$  clusters. Each point in the data set belongs to exactly one cluster in exactly one set. The agglomerative clustering then iteratively combines cluster sets such that in each iteration two sets are combined to one set with  $K$  clusters.

**Agglomerative Clustering** Finally, the *AgglomerativeClustering* function builds a hierarchy of cluster sets, by iteratively combining cluster set pairs. In each iteration it computes the similarity between any such pair, defined to be the lowest similarity between their cluster members, which is in turn defined to be the lowest cosine similarity between their point members. The most similar cluster sets are combined such that each of the clusters in one set is mapped to its most similar cluster in the other set. In this step the algorithm generates data partitions at different granularity levels from finest (from the iterative sampling step) to the coarsest set (generated by the last agglomerative clustering iteration and consisting of exactly  $K$  clusters). This property is useful as the optimal level of generalization may be task dependent.

## 4 Evaluation

**Data sets and gold standards** We evaluated the SCFs and verb clusters on gold standard datasets. We based our set of the largest available joint set for SCFs and VCs - that of (de Cruys et al., 2012). It provides SCF annotations for 183 verbs (an average of 12.3 SCF types per verb) obtained by annotating 250 corpus occurrences per verb with the SCF types of (de Cruys et al., 2012). The verbs represent a range of Levin classes at the top level of the hierarchy in VerbNet (Kipper-Schuler, 2005). Where a verb has more than one VerbNet class, we assign it to the one supported by the highest number of member verbs. To ensure suf-

Model	C  = 20, 21.6			C  = 40, 41			C  = 60, 58.6			C  = 69, 77.6			C  = 89, 97.4		
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
DPP-cluster	93.1	17.3	29.3	77.9	25.4	<b>38.3</b>	63	31.9	<b>42.3</b>	43.8	33.6	<b>38.1</b>	34.4	40.6	<b>37.2</b>
AC	67	17.8	28.2	46.6	24	31.7	40.5	29.4	34	33	34.9	33.9	24.7	41.1	30.9
SC	32.1	27.5	<b>29.6</b>	26.6	35.9	30.6	23.7	41.5	30.2	22.8	43.6	29.9	21.6	48.7	29.9

Table 1: Verb clustering evaluation for the last five iterations of our DPP-cluster model and the baseline agglomerative clustering algorithm (AC, see text for its description), and for the spectral clustering (SC) algorithm of (Sun and Korhonen, 2009) with the same number of clusters induced by DPP-cluster.  $|C|$  is the number of clusters for DPP-cluster and SC (first number) and for AC (second number). The F-score performance of DPP-cluster is superior in 4 out of 5 cases.

Arg. per verb	P (DPP)	P (AC)	P (B)	P (NF)	R (DPP)	R (AC)	R (B)	R (NF)	ERR DPP	ERR AC	ERR B
$\leq 200$ (133 verbs)	27.3	23.7	27.3	23.1	9.9	7.6	8	11.3	<b>3.4</b>	0.16	1.55
$\leq 600$ (205 verbs)	26.5	25	27.3	22.6	14.8	11.5	11.9	16.6	<b>2.3</b>	0.50	1.1
$\leq 1000$ (238 verbs)	24.6	23.6	25.6	21.1	17.5	13.8	14.7	19.8	<b>1.6</b>	0.42	0.95

Table 2: Performance of the Corpus Statistics SP baseline (non-filtered, NF) as well as for three filtering methods: frequency based (filter-baseline, B), DPP-cluster based (DPP) and AC cluster based (AC). P (method) and R (method) present the precision and recall of the method respectively. The error reduction ratio (ERR) is the ratio between the reduction in precision error achieved by each method and the increase in recall error (each method is compared to the NF baseline). Ratio greater than 1 means that the reduction in precision error is larger than the increase in recall error (see text for exact definition). DPP based filtering provides substantially better ratio.

efficient representation of each class, we collected from VerbNet the verbs for which at least one of the possible classes is represented in the 183 verbs set by at least one and at most seven verbs. This yielded 101 additional verbs which we added to the gold standard with the initial 183 verbs.

We parsed the BNC corpus with the RASP parser (Briscoe et al., 2006) and used it for feature extraction. Since 176 out of the 183 initial verbs are represented in this corpus, our final gold standard consists of 34 classes containing 277 verbs, of which 176 have SCF gold standard and has been evaluated for this task. We set the parameters of our algorithm on an held-out data, consisting of different verbs than those used in our experiments, to be  $M = 10000$ ,  $K = 20$  and  $T = 10$ .

**Clustering Evaluation** We first evaluate the quality of the clusters induced by our algorithm (DPP-cluster) compared to the gold standard VCs (table 1). To evaluate the importance of the DPP component, we compare to the performance of a version of our algorithm where everything is kept fixed except from the sampling which is done from a uniform distribution rather than from the DPP joint kernel (this model is denoted in the table with AC for agglomerative clustering)<sup>4</sup>. We also compare to the state-of-the-art spectral clustering method of Sun and Korhonen (2009) where our

<sup>4</sup>Importantly, the kernel matrix  $L$  used in the agglomerative clustering process is also used by AC.

kernel matrix is used for the distance between data points (SC)<sup>5</sup>.

We evaluated the unified cluster set induced in each iteration of our algorithm and of the AC baseline and induced the same number of clusters as in each iteration of our algorithm using the SC baseline. Since the number of clusters in each iteration is not an argument for our algorithm or for the AC baseline, the number of clusters slightly differ between the two. The AC and SC baseline results were averaged over 5 and 100 runs respectively. DPP-cluster has produced identical output across runs.

The table demonstrates the superiority of the DPP-cluster model. For four out of five conditions its F-score performance outperforms the baselines by 4.2-8.3%. Moreover, in all conditions its recall performances are substantially higher than those of the baselines (by 9.7-26.1%). Note that DPP-cluster runs for 17 iterations while the AC baseline performs only 6. We therefore evaluated only the last 5 iterations of each model<sup>6</sup>.

**SCF evaluation** For this evaluation, we first built a baseline SCF lexicon based on the parsed

<sup>5</sup>Sun and Korhonen (2009) report better results than those we report for their algorithm (on a different data set). Note, however, that they used the output of a rule-based SCF system as a source of features, as opposed to our unsupervised approach.

<sup>6</sup>For the additional comparable iteration the result pattern is very similar to the ( $C = 89, 97.4$ ) case in the table, and is not presented due to space limitations.

---

**Algorithm 1** The DPP-cluster clustering algorithm.  $K$  is the size of the sampled subsets,  $M$  is the number of subsets sampled at each iteration,  $\mathcal{Y}$  is the verb set,  $T$  is the number of most probable samples to be used in each iteration

---

**Algorithm DPP-cluster :**

Arguments:  $K, M, \mathcal{Y}, T$

Return: cluster sets  $S = \{S_1, \dots, S_n\}$

$i \leftarrow 1$

$S \leftarrow \emptyset$

**while**  $\mathcal{Y} \neq \emptyset$  **do**

$(L^1, S^1) \leftarrow \text{GenKernelMatrix}(\mathcal{Y}, SCF)$

$(L^2, S^2) \leftarrow \text{GenKernelMatrix}(\mathcal{Y}, SP)$

$(L^{12}, S^{12}) \leftarrow \text{GenJointMat}(L^1, L^2)$

$\text{samples} \leftarrow \text{sampleDpp}(L, K, M)$

$\text{topSamples} \leftarrow \text{exTop}(\text{samples}, T)$

$S_i \leftarrow \text{cluster}(\text{topSamples}, L)$

$\mathcal{Y} \leftarrow \mathcal{Y} - \text{elements}(S_i)$

$S \leftarrow S \cup S_i$

$i \leftarrow i + 1$

**end while**

*AgglomerativeClustering*( $S$ )

---

**Function cluster :**

Arguments:  $\text{topSamples}, L$

Return:  $S$

$S \leftarrow \emptyset, \text{topSample} \leftarrow \emptyset$

$i \leftarrow 1$

**while**  $(\text{topSample} \cap \text{elements}(S) = \emptyset)$  **do**

$\text{topSample} \leftarrow \text{topSamples}(i)$

$S \leftarrow \text{m1Mapping}(\text{topSample}, S)$

$i \leftarrow i + 1$

**if**  $(i > \text{size}(\text{topSamples}))$  **then**

**return**  $S$

**end if**

**end while**

---

BNC corpus. We do this by gathering the GR combinations for each of the verbs in our gold standard, assuming they are frames and gathering their frequencies. Note that this *corpus statistics* baseline is a very strong baseline that performs very similarly to (de Cruys et al., 2012), the best unsupervised SCF model we are aware of, when run on their dataset<sup>7</sup>.

As shown in table 3 the corpus statistics baseline achieves high recall (84%) at the cost of low precision (52.5%) (similar pattern has been

---

<sup>7</sup>personal communication with the authors.

demonstrated for the system of de Cruys et al. (2012)). On the other extreme, two other commonly used baselines strongly prefer precision. These are the Most Frequent SCF (O’Donovan et al., 2005) which uniformly assigns to all verbs the two most frequent SCFs in general language, transitive (SUBJ-DOBJ) and intransitive (SUBJ) (and results in poor F-score), and a filtering that removes frames with low corpus frequencies (which results in low recall even when trying to provide the maximum recall for a given precision level). The task we address is therefore to improve the precision of the corpus statistics baseline in a way that does not substantially harm the F-score.

To remedy this imbalance, we apply a cluster based filtering method on top of the maximum-recall frequency filter. This filter excludes a candidate frame from a verb’s lexicon only if it meets the frequency filter criterion and appears in no more than  $N$  other members of the cluster of the verb in question. The filter utilizes the clustering produced by the seventh to last iteration of DPP-cluster that contains seven clusters with approximately 30 members each. Such clustering should provide a good generalization level for the task.

We report results for moderate as well as aggressive filtering ( $N = 3$  and  $N = 7$  respectively). Table 3 clearly demonstrates that cluster based filtering (DPP-cluster and AC) is the only method that provides a good balance between the recall and the precision of the SCF lexicon. Moreover, the lexicon induced by this method includes a substantially higher number of frames per verb compared to the other filtering methods. While both AC and DPP-cluster still prefer recall to precision, DPP-cluster does so to a smaller extent<sup>8</sup>. This clearly demonstrates that the clustering serves to provide SCF acquisition with semantic information needed for improved performance.

**SP evaluation** We explore a variant of the pseudo-disambiguation task of Rooth et al. (1999) which has been applied to SP acquisition by a number of recent papers (e.g. (de Cruys et al., 2012)). Rooth et al. (1999) proposed to judge which of two verbs  $v$  and  $\tilde{v}$  is more likely to take a given noun  $n$  as its argument. In their experiments the model has to choose between a pair  $(v, n)$  that

---

<sup>8</sup>We show results for the maximum recall frequency filtering with precision equals to 80 or 90. When the frequency threshold is further reduced from 0.03, the same result pattern hold. We do not give a detailed description due to space limitations.

		Corpus Statistics: [P = 52.5, R = 84, F = 64.6, AF = 12.3] Most Frequent SCF: [P = 86.7, R = 22.5, F = 35.8, AF = 2]							
		Clustering Moderate				Clustering Aggressive			
Maximum Recall Frequency Threshold	Model	P	R	F	AF	P	R	F	AF
threshold = 0.03, Prec. > 80 [P=88.7,R=52.4,F=65.9,AF=4.5]	DPP-cluster	60.8	68.3	64.3	8.7	64.1	64.2	64.2	7.7
	AC	58	73.2	64.6	9.7	61.3	68.9	64.7	8.6
threshold = 0.05, Prec. > 90 [P=92.3,R=44.4,F=59.9,AF=3.7]	DPP-cluster	60.1	64.6	62.3	8.7	63.3	59.3	61.3	7.2
	AC	57.5	70.6	63.2	9.4	60.7	65.4	62.7	8.3

Table 3: SCF Results for the DPP-cluster model compared to the Corpus Statistics baseline, Most Frequent SCF baseline, maximum-recall frequency thresholding with the maximum threshold values that keep precision above 80 (threshold = 0.03) and above 90 (threshold = 0.05), and the AC clustering baseline. AF is the average number of frames per verb. **All methods except from cluster based filtering (DPP-cluster and AC) induce lexicons with strong recall/precision imbalance. Cluster based filtering keeps a larger number of frames in the lexicon compared to the frequency thresholding baseline, while keeping similar F-score levels.** DPP-cluster provides better recall/precision balance than AC.

appears only in the test corpus and a pair  $(\tilde{v}, n)$  that appears neither in the test nor in the training corpus. Note, however, that this test only evaluates the capability of a model to distinguish a correct unseen verb-argument pair from an incorrect one, but not its capability to identify erroneous pairs when no alternative pair is presented. This last property can strongly affect the precision of the model.

We therefore propose to measure both aspects of the SP task by computing both the recall and the precision between the list of possible arguments a verb can take according to the model and the corresponding test corpus list<sup>9</sup>.

We evaluate the value of our clustering for SP acquisition in the particularly challenging scenario of domain adaptation. For each of the verbs in our set we induce a list of possible noun direct objects from the BNC corpus and an equivalent list from the North American News Text (NANT) corpus. Following previous work (e.g. (de Cruys et al., 2012)) arguments are identified using a parser (RASP in our case). Using the verb clusters we create a filtered version of the BNC argument lexicon which includes in the noun argument list of a verb only those nouns that appear in the BNC as arguments of that verb and of one of its cluster members. For each verb we then compare the filtered as well as the non-filtered BNC induced lexicon to the NANT lexicon by computing the average recall and precision between the argument lists

<sup>9</sup>In principle these measures can take into account the probability assigned by the model to each argument and the corresponding test corpus frequency. In this work we compute probability-ignorant scores and keep more sophisticated evaluations for future research.

and then report the average scores across all verbs. We compare to a baseline which maintains only noun arguments that appear at least twice in BNC<sup>10</sup>. As a final measure of performance we compute the ratio between the reduction in precision error (i.e.  $\frac{p_{model} - p_{baseline}}{100 - p_{baseline}}$ ) and the increase in recall error ( $\frac{r_{baseline} - r_{model}}{100 - r_{model}}$ ).

Table 2 presents the results for verbs with up to 200, 600 and 1000 noun arguments in the training data. In all cases, the relative error reduction of the DPP cluster filter is substantially higher than that of the frequency baseline. Note that for this task the baseline AC clusters are of low quality which is reflected by an error reduction ratio of up to 0.5.

## 5 Conclusions and Future Work

In this paper we have presented the first unified framework for the induction of verb clusters, sub-categorization frames and selectional preferences from corpus data. Our key idea is to cluster together verbs with similar SCFs and SPs and to use the resulting clusters for SCF and SP induction. To implement our idea we presented a novel method which involves constructing a product DPP model for SCFs and SPs and introduced a new algorithm that utilizes the efficient DPP sampling algorithms to cluster together verbs with similar SCFs and SPs. The induced clusters performed well in evaluation against a VerbNet -based gold standard and proved useful in improving the quality of SCFs and SPs over strong baselines.

Our results demonstrate the benefits of a unified framework for acquiring lexical informa-

<sup>10</sup>we experimented with other threshold values for this baseline but the recall in those case becomes very low.

tion about different aspects of verbal predicate-argument structure. Not only the acquisition of different types information (syntactic and semantic) can support and inform each other, but also a unified framework can be useful for NLP tasks and applications which require rich information about predicate-argument structure. In future work we plan to apply our approach on larger scale data sets and gold standards and to evaluate it in different domains, languages and in the context of NLP tasks such as syntactic parsing and SRL.

In addition, in our current framework SCF and SP information is used for clustering which is in turn used to improve SCF and SP quality. At this stage no further information flows from the SCF and SP models to the clustering model. A natural extension of our unified framework is to construct a joint model in which the predictions for all three tasks inform each other at all stages of the prediction process.

## Acknowledgements

The work in this paper was funded by the Royal Society University Research Fellowship (UK).

## References

- Ivana Romina Altamirano and Laura Alonso i Alemany. 2010. IRASubcat, a highly customizable, language independent tool for the acquisition of verbal subcategorization information from corpus. In *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas*.
- Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *COLING-ACL-98*.
- Roberto Basili, Diego De Cao, Paolo Marocco, and Marco Pennacchiotti. 2007. Learning selectional preferences for entailment or paraphrasing rules. In *RANLP 2007*, Borovets, Bulgaria.
- Rahul Bhagat, Patrick Pantel, and Eduard Hovy. 2007. Ledir: An unsupervised algorithm for learning directionality of inference rules. In *EMNLP-07*, page 161170, Prague, Czech Republic.
- Akshar Bharati, Sriram Venkatapathy, and Prashanth Reddy. 2005. Inferring semantic roles using subcategorization frames and maximum entropy model. In *CoNLL-05*.
- Ted Briscoe and John Carroll. 1997. Automatic extraction of subcategorization from corpora. In *ANLP-97*.
- E.J. Briscoe, J. Carroll, and R. Watson. 2006. The second release of the rasp system. In *COLING/ACL interactive presentation session*.
- Glenn Carroll and Mats Rooth. 1996. Valence induction with a head-lexicalized pcfg. In *EMNLP-96*.
- Paula Chesley and Susanne Salmon-Alt. 2006. Automatic extraction of subcategorization frames for french. In *LREC-06*.
- Kostadin Cholakov and Gertjan van Noord. 2010. Using unknown word techniques to learn known words. In *EMNLP-10*.
- Hoa Trang Dang. 2004. *Investigations into the Role of Lexical Semantics in Word Sense Disambiguation*. Ph.D. thesis, CIS, University of Pennsylvania.
- Tim Van de Cruys, Laura Rimell, Thierry Poibeau, and Anna Korhonen. 2012. Multi-way tensor factorization for unsupervised lexical acquisition. In *COLING-12*.
- Lukasz Dkebowksi. 2009. Valence extraction using EM selection and co-occurrence matrices. *Language resources and evaluation*, 43(4):301–327.
- Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *ACL 2007*, Prague, Czech Republic.
- J. Gillenwater, A. Kulesza, and B. Taskar. 2012. Discovering diverse and salient threads in document collections. In *EMNLP-12*.
- Ralph Grishman, Catherine Macleod, and Adam Meyers. 1994. Complex syntax: Building a computational lexicon. In *COLNIG-94*.
- G.E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800.
- Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In *Proceedings Of NAACL-HLT-06 short papers*.
- Dino Ienco, Serena Villata, and Cristina Bosco. 2008. Automatic extraction of subcategorization frames for italian. In *LREC-08*.
- Eric Joanis, Suzanne Stevenson, and David James. 2008. A general feature space for automatic verb classification. *Natural Language Engineering*.
- Daisuke Kawahara and Sadao Kurohashi. 2010. Acquiring reliable predicate-argument structures from raw corpora for case frame compilation. In *LREC-10*.
- Karin Kipper-Schuler. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, June.

- Anna Korhonen, Yuval Krymolowski, and Nigel Collier. 2008. The choice of features for classification of verbs in biomedical texts. In *Proceedings of COLING-08*.
- Anna Korhonen. 2002. Semantically motivated subcategorization acquisition. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition-Volume 9*.
- A. Kulesza and B. Taskar. 2010. Structured determinantal point processes. In *NIPS-10*.
- A. Kulesza and B. Taskar. 2012a. k-dpps: fixed-size determinantal point processes. In *ICML-11*.
- A. Kulesza and B. Taskar. 2012b. Learning determinantal point processes. In *UAI-12*.
- Alex Kulesza and Ben Taskar. 2012c. Determinantal point processes for machine learning. In *arXiv:1207.6083*.
- A. Kulesza. 2012. *Learning with determinantal point processes*. Ph.D. thesis, CIS, University of Pennsylvania.
- Alessandro Lenci, Barbara McGillivray, Simonetta Montemagni, and Vito Pirrelli. 2008. Unsupervised acquisition of verb subcategorization frames from shallow-parsed corpora. In *LREC-08*.
- Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. Chicago, IL.
- Jianguo Li and Chris Brew. 2008. Which are the best features for automatic verb classification. In *ACL-08*.
- Tom Lippincott, Anna Korhonen, and Diarmuid Ó Séaghdha. 2012. Learning syntactic verb frames using graphical models. In *ACL-12*, Jeju, Korea.
- Cédric Messiant, Anna Korhonen, and Thierry Poibeau. 2008. LexSchem: A large subcategorization lexicon for French verbs. In *LREC-08*.
- George A. Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- Alessandro Moschitti and Roberto Basili. 2005. Verb subcategorization kernels for automatic semantic labeling. In *Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*.
- Ruth O’Donovan, Michael Burke, Aoife Cahill, Josef van Genabith, and Andy Way. 2005. Large-scale induction and evaluation of lexical resources from the penn-ii and penn-iii treebanks. *Computational Linguistics*, 31:328–365.
- Diarmuid Ó Séaghdha and Anna Korhonen. 2011. Probabilistic models of similarity in syntactic context. In *EMNLP-11*, Edinburgh, UK.
- Diarmuid Ó Séaghdha. 2010. Latent variable models of selectional preference. In *ACL-10*, Uppsala, Sweden.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.
- Judita Preiss, Ted Briscoe, and Anna Korhonen. 2007. A system for large-scale acquisition of verbal, nominal and adjectival subcategorization frames from corpora. In *ACL-07*.
- Joseph Reisinger and Raymond Mooney. 2011. Cross-cutting models of lexical semantics. In *EMNLP-11*, Edinburgh, UK.
- Alan Ritter and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *ACL-10*.
- Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a semantically annotated lexicon via em-based clustering. In *ACL-99*.
- Karin Kipper Schuler. 2006. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.
- S. Schulte im Walde, C. Hying, C. Scheible, and H. Schmid. 2008. Combining EM training and the MDL principle for an automatic verb classification incorporating selectional preferences. In *ACL-08*, pages 496–504.
- Sabine Schulte im Walde. 2006. Experiments on the automatic induction of german semantic verb classes. *Computational Linguistics*, 32(2):159–194.
- Lei Shi and Rada Mihalcea. 2005. Putting pieces together: Combining framenet, verbnet and wordnet for robust semantic parsing. In *CICLING-05*.
- Lin Sun and Anna Korhonen. 2009. Improving verb clustering with automatically acquired selectional preferences. In *EMNLP-09*, Singapore.
- Lin Sun and Anna Korhonen. 2011. Hierarchical verb clustering using graph factorization. In *EMNLP-11*.
- Lin Sun, Anna Korhonen, and Yuval Krymolowski. 2008. Verb class discovery from rich syntactic data. *Lecture Notes in Computer Science*, 4919(16).
- Robert Swier and Suzanne Stevenson. 2004. Unsupervised semantic role labelling. In *EMNLP-04*.
- Stefan Thater, Hagen Furstenu, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *ACL-10*, Uppsala, Sweden.
- Tim Van de Cruys. 2009. A non-negative tensor factorization model for selectional preference induction. In *Proceedings of the workshop on Geometric Models for Natural Language Semantics (GEMS)*.

- Andreas Vlachos, Anna Korhonen, and Zoubin Ghahramani. 2009. Unsupervised and constrained dirichlet process mixture models for verb clustering. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*.
2008. *Robustness and generalization of role sets: PropBank vs. VerbNet*.
- Benat Zafirain, Eneko Agirre, and Lluís Marquex. 2009. Generalizing over lexical features: Selectional preferences for semantic role classification. In *ACL-IJCNLP-09*, Singapore.
- Guangyou Zhou, Jun Zhao, Kang Liu, and Li Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *ACL-11*, Portland, OR.

# Semantic Frames to Predict Stock Price Movement

Boyi Xie, Rebecca J. Passonneau, Leon Wu

Center for Computational Learning Systems  
Columbia University  
New York, NY USA

(bx2109|becky|leon.wu)@columbia.edu

Germán G. Creamer

Howe School of Technology Management  
Stevens Institute of Technology  
Hoboken, NJ USA

gcreamer@stevens.edu

## Abstract

Semantic frames are a rich linguistic resource. There has been much work on semantic frame parsers, but less that applies them to general NLP problems. We address a task to predict change in stock price from financial news. Semantic frames help to generalize from specific sentences to scenarios, and to detect the (positive or negative) roles of specific companies. We introduce a novel tree representation, and use it to train predictive models with tree kernels using support vector machines. Our experiments test multiple text representations on two binary classification tasks, change of price and polarity. Experiments show that features derived from semantic frame parsing have significantly better performance across years on the polarity task.

## 1 Introduction

A growing literature evaluates the financial effects of media on the market (Tetlock, 2007; Engelberg and Parsons, 2011). Recent work has applied NLP techniques to various financial media (conventional news, tweets) to detect sentiment in conventional news (Devitt and Ahmad, 2007; Haider and Mehrotra, 2011) or message boards (Chua et al., 2009), or discriminate expert from non-expert investors in financial tweets (Bar-Haim et al., 2011). With the exception of Bar-Haim et al. (2011), these NLP studies have relied on small corpora of hand-labeled data for training or evaluation, and the connection to market events is done indirectly through sentiment detection. We hypothesize that conventional news can be used to predict changes in the stock price of specific companies, and that the semantic features that best represent relevant aspects of the news vary across

On Wednesday, April 11th, 2012, **Google Inc** announced its first **quarterly earnings** report, a week before the April 20 options contracts expiration in *contrast* to its history of reporting a day before monthly options expirations. The **stock price of Google** *surged* 3.85% from April 10th's \$626.86 to 12th's \$651.01. On Friday, April 13th, news reported **Oracle Corp** would *sue* **Google Inc**, claiming **Google's Android operating system** *tramples* **its intellectual property rights**. Jury selection was set for the next Monday. **Google's stock price** *tumbled* 4.06% on Friday, and continued to drop in the following week.

Figure 1: Summary of financial news items pertaining to *Google* in April, 2012.

market sectors. To test this hypothesis, we use price information to label data from six years of financial news. Our experiments test several document representations for two binary classification tasks, change of price and polarity. Our main contribution is a novel tree representation based on semantic frame parses that performs significantly better than enriched bag-of-words vectors.

Figure 1 shows a constructed example based on extracts from financial news about *Google* in April, 2012. It illustrates how a series of events reported in the news precedes and potentially predicts a large change in *Google's* stock price. *Google's* early announcement of quarterly earnings possibly presages trouble, and its stock price falls soon after reports of a legal action against *Google* by *Oracle*. To produce a coherent story, the original sentences were edited for Figure 1, but they are in the style of actual sentences from our dataset. Accurate detection of events and relations that might have an impact on stock price should benefit from document representation that captures sentiment in lexical items (e.g., *aggressive*) combined with the conceptual relations captured by FrameNet (Ruppenhofer and Rehbein, 2012). A frame is a lexical semantic representa-

tion of the conceptual roles played by parts of a clause, and relates different lexical items (e.g., *report*, *announce*) to the same situation type. In the figure, some of the words that evoke frames have been underlined, and role fillers are outlined by boxes or ovals. Sentiment words are in italics.

To the best of our knowledge, this paper is the first to apply semantic frames in this domain. On the polarity task, the semantic frame features encoded as trees perform significantly better across years and sectors than bag-of-words vectors (BOW), and outperform BOW vectors enhanced with semantic frame features, and a supervised topic modeling approach. The results on the price change task show the same trend, but are not statistically significant, possibly due to the volatility of the market in 2007 and the following several years. Yet even modest predictive performance on both tasks could have an impact, as discussed below, if incorporated into financial models such as Rydberg and Shephard (2003). We first discuss the motivation and related work. Section 4 presents vector-based and tree-based features from semantic frame parses, and section 5 describes our dataset. The experimental design and results appear in the following section, followed by discussion and conclusions.

## 2 Motivation

Financial news is a rich vein for NLP applications to mine. Many news organizations that feature financial news, such as Reuters, the Wall Street Journal and Bloomberg, devote significant resources to the analysis of corporate news.

Much of the data that would support studies of a link between the news media and the market are publicly available. As pointed out by Tetlock et al. (2008), linguistic communication is a potentially important source of information about firms' fundamental values. Because very few stock market investors directly observe firms' production activities, they get most of their information secondhand. Their three main sources are analysts' forecasts, quantifiable publicly disclosed accounting variables, and descriptions of firms' current and future profit-generating activities. If analyst and accounting variables are incomplete or biased measures of firms' fundamental values, linguistic variables may have incremental explanatory power for firms' future earnings and returns.

Consider the following sentences:

*Oracle sued Google in August 2010, saying Google's Android mobile operating system infringes its copyrights and patents for the Java programming language.* (a)

*Oracle has accused Google of violating its intellectual property rights to the Java programming language.* (b)

*Oracle has blamed Google and alleged that the latter has committed copyright infringement related to Java programming language held by Oracle.* (c)

*Oracle's Ellison says couldn't sway Google on Java.* (d)

Sentences *a*, *b* and *c* are semantically similar, but lexically rather distinct: the shared words are the company names and *Java (programming language)*. Bag-of-Words (BOW) document representation is difficult to surpass for many document classification tasks, but cannot capture the degree of semantic similarity among these sentences. Methods that have proven successful for paraphrase detection (Deerwester et al., 1990; Dolan et al., 2004), as in the main clauses of *b* and *c*, include latent variable models that simultaneously capture the semantics of words and sentences, such as latent semantic analysis (LSA) or latent Dirichlet allocation (LDA). However, our task goes beyond paraphrase detection. The first three sentences all indicate an adversarial relation of *Oracle* to *Google* involving a negative judgement. It would be useful to capture the similarities among all three of these sentences, and to distinguish the role of each company (who is suing and who is being sued). Further, these three sentences potentially have a greater impact on market perception of *Google* in contrast to a sentence like *d*, that refers to the same conflict more indirectly, and whose main clause verb is *say*. We hypothesize that semantic frames can address these issues.

Most of the NLP literature on semantic frames addresses how to build robust semantic frame parsers, with intrinsic evaluation against gold standard parses. There have been few applications of semantic frame parsing for extrinsic tasks. To test for measurable benefits of semantic frame parsing, this paper poses the following questions:

1. Are semantic frames useful for document representation of financial news?
2. What aspects of frames are most useful?
3. What is the relative performance of document representation that relies on frames?

4. What improvements could be made to best exploit semantic frames?

Our work is not aimed at investment profit. Rather, we investigate whether computational linguistic methodologies can improve our understanding of a company’s fundamental market value, and whether linguistic information derived from news produces a consistent enough result to benefit more comprehensive financial models.

### 3 Related Work

NLP has recently been applied to financial text for market analysis, primarily using bag-of-words (BOW) document representation. Luss and d’Aspremont (2008) use text classification to model price movements of financial assets on a per-day basis. They try to predict the direction of return, and abnormal returns, defined as an absolute return greater than a predefined threshold. Kogan et al. (2009) address a text regression problem to predict the financial risk of investment in companies. They analyze 10-K reports to predict stock return volatility. They also predict whether a company will be delisted following its 10-K report. Ruiz et al. (2012) correlate text with financial time series volume and price data. They find that graph centrality measures like page rank and degree are more strongly correlated to both price and traded volume for an aggregation of similar companies, while individual stocks are less correlated. Lavrenko et al. (2000) present an approach to identify news stories that influence the behavior of financial markets, and predict trends in stock prices based on the content of news stories that precede the trends. Luss and d’Aspremont (2008) and Lavrenko et al. (2000) both point out the desire for document feature engineering as future research directions. We explore a rich feature space that relies on frame semantic parsing.

Sentiment analysis figures strongly in NLP work on news. General Inquirer (GI), a content analysis program, is used to quantify pessimism of news in Tetlock (2007) and Tetlock et al. (2008). Other resources for sentiment detection include the Dictionary of Affect in Language (DAL) to score the prior polarity of words, as in Agarwal et al. (2011) on social media data. Our study incorporates DAL scores along with other features.

FrameNet is a rich lexical resource (Fillmore et al., 2003), based on the theory of frame semantics (Fillmore, 1976). There is active research

Category	Features	Value type
Frame attributes	F, FT, FE wF, wFT, wFE	$\mathbb{N}$ $\mathbb{R}_{\geq 0}$
BOW	UniG, BiG, TriG wUniG, wBiG, wTriG	$\mathbb{N}$ $\mathbb{R}_{\geq 0}$
pDAL	all-Pls, all-Act, all-Img VB-Pls, VB-Act, VB-Img JJ-Pls, JJ-Act, JJ-Img RB-Pls, RB-Act, RB-Img	$\mathbb{R}_{\sim\mu=0, std=1}$ $\mathbb{R}_{\sim\mu=0, std=1}$ $\mathbb{R}_{\sim\mu=0, std=1}$ $\mathbb{R}_{\sim\mu=0, std=1}$

Table 1: FWD features (Frame, bag-of-Words, part-of-speech DAL score) and their value types.

to build more accurate parsers (Das and Smith, 2011; Das and Smith, 2012). Semantic role labeling using FrameNet has been used to identify an opinion with its holder and topic (Kim and Hovy, 2006). For deep representation of sentiment analysis, Ruppenhofer and Rehbein (2012) propose SentiFrameNet.

Our work addresses classification tasks that have potential relevance to an influential financial model (Rydberg and Shephard, 2003). This model decomposes stock price analysis of financial data into a three-part ADS model - *activity* (a binary process modeling the price move or not), *direction* (another binary process modeling the direction of the moves) and *size* (a number quantifying the size of the moves). Our two binary classification tasks for news, price change and polarity, are analogous to their *activity* and *direction*. In contrast to the ADS model, our approach does not calculate the conditional probability of each factor. At present, our goal is limited to the determination of whether NLP features can uncover information from news that could help predict stock price movement or support analysts’ investigations.

### 4 Methods

We propose two approaches for the use of semantic frames. The first is a rich vector space based on semantic frames, word forms and DAL affect scores. The second is a tree representation that encodes semantic frame features, and depends on tree kernel measures for support vector machine classification. The semantic parses of both methods are derived from SEMAFOR<sup>1</sup> (Das and Smith, 2012; Das and Smith, 2011), which solves the semantic parsing problem by rule-based target identification, log-linear model based frame identification and frame element filling.

<sup>1</sup><http://www.ark.cs.cmu.edu/SEMAFOR>.

Frame (F)	Judgment_comm.	Commerce_buy
Target (FT)	accuse sue charge	buy purchase bid
Frame Element (FE)	COMMUNICATOR EVALUEE REASON	BUYER SELLER GOODS

Table 2: Sample frames.

#### 4.1 Semantic Frame based FWD Features

Table 1 lists 24 types of features, including semantic Frame attributes, bag-of-Words, and scores for words in the Dictionary of Affect in Language by part of speech (pDAL). We refer to these features as **FWD** features throughout the paper. FWD features are used alone and in combinations.

FrameNet defines hundreds of frames, each of which represents a scenario associated with semantic roles, or frame elements, that serve as participants in the scenario the frame signifies. Table 2 shows two frames. The frame *Judgment\_communication* (*JC* or *Judgment\_comm.* in the rest of the paper) represents a scenario in which a COMMUNICATOR communicates a judgment of an EVALUEE for some REASON. It is evoked by (target) words such as *accuse* or *sue*.

Here we use **F** for the frame name, **FT** for the target words, and **FE** for frame elements. We use both frequency and weighted scores. For example, we define *idf*-adjusted weighted frame features, such as  $wF$  for attribute  $F$  in document  $d$  as  $wF_{F,d} = f(F, d) \times \log \frac{|D|}{|d \in D: F \in d|}$ , where  $f(F, d)$  is the frequency of frame  $F$  in  $d$ ,  $D$  is the whole document set and  $|\cdot|$  is the cardinality operator.

Bag-of-Words features include term frequency and *tfidf* of unigrams, bigrams, and trigrams.

DAL (Dictionary of Affect in Language) is a psycholinguistic resource to measure the emotional meaning of words and texts (Whissel, 1989). It includes 8,742 words that were annotated for three dimensions: Pleasantness (Pls), Activation (Act), and Imagery (Img). Agarwal et al. (2009) introduced part-of-speech specific DAL features for sentiment analysis. We follow their approach by averaging the scores for all words, verb only, adjective only, and adverb only words. Feature values are normalized to mean of zero and standard deviation of one.

#### 4.2 SemTree Feature Space and Kernels

We propose **SemTree** as another feature space to encode semantic information in trees. SemTree

can distinguish the roles of each company of interest, or *designated object* (e.g. who is suing and who is being sued).

##### 4.2.1 Construction of Tree Representation

The semantic frame parse of a sentence is a forest of trees, each of which corresponds to a semantic frame. SemTree encodes the original frame structure and its leaf words and phrases, and highlights a designated object at a particular node as follows. For each lexical item (target) that evokes a frame, a backbone is found by extracting the path from the root to the role filler mentioning a designated object; the backbone is then reversed to promote the designated object. If multiple frames have been assigned to the same designated object, their backbones are merged. Lastly, the frame elements and frame targets are inserted at the frame root.

The top of Figure 2 shows the semantic parse for sentence  $a$  from section 2; we use it to illustrate tree construction for designated object *Oracle*. The parse has two frames (Figure 2-(1)&(2)), one corresponding to the main clause (verb *sue*), and the other for the tenseless adjunct (verb *say*). The reversed paths extracted from each frame root to the designated object *Oracle* become the backbones (Figures 2-(3)&(4)). After merging the two backbones we get the resulting SemTree, as shown in Figure 2-(5). By the same steps, this sentence would also yield a SemTree with *Google* at the root, in the role of EVALUEE.

##### 4.2.2 Kernels and Tree Substructures

The tree kernel (Moschitti, 2006; Collins and Duffy, 2002) is a function of tree similarity, based on common substructures (tree fragments). There are two types of substructures. A subtree (ST) is defined as any node of a tree along with all its descendants. A subset tree (SST) is defined as any node along with its immediate children and, optionally, part or all of the children’s descendants. Each tree is represented by a  $d$  dimensional vector where the  $i$ ’th component counts the number of occurrences of the  $i$ ’th tree fragment.

Define the function  $h_i(T)$  as the number of occurrences of the  $i$ ’th tree fragment in tree  $T$ , so that  $T$  is now represented as  $\mathbf{h}(T) = (h_1(T), h_2(T), \dots, h_d(T))$ . We define the set of nodes in tree  $T_1$  and  $T_2$  as  $N_{T_1}$  and  $N_{T_2}$  respectively. We define the indicator function  $I_i(n)$  to be 1 if subtree  $i$  is seen rooted at node  $n$ , and 0 otherwise. It follows that  $h_i(T_1) = \sum_{n_1 \in N_{T_1}} I_i(n_1)$

Designated object: Oracle (ORCL)  
Sentence: Oracle sued Google in August 2010, saying Google’s Android mobile operating system infringes its copyrights and patents for the Java programming language.  
SRL: [Oracle<sub>JC.FE.Communicator</sub>, Stmt.FE.Speaker] [sued<sub>JC.Target</sub>] [Google<sub>JC.FE.Evaluee</sub>] in August 2010, [saying<sub>Stmt.Target</sub>] [Google’s Android mobile operating system infringes its copyrights and patents for the Java programming language<sub>Stmt.FE.Message</sub>].

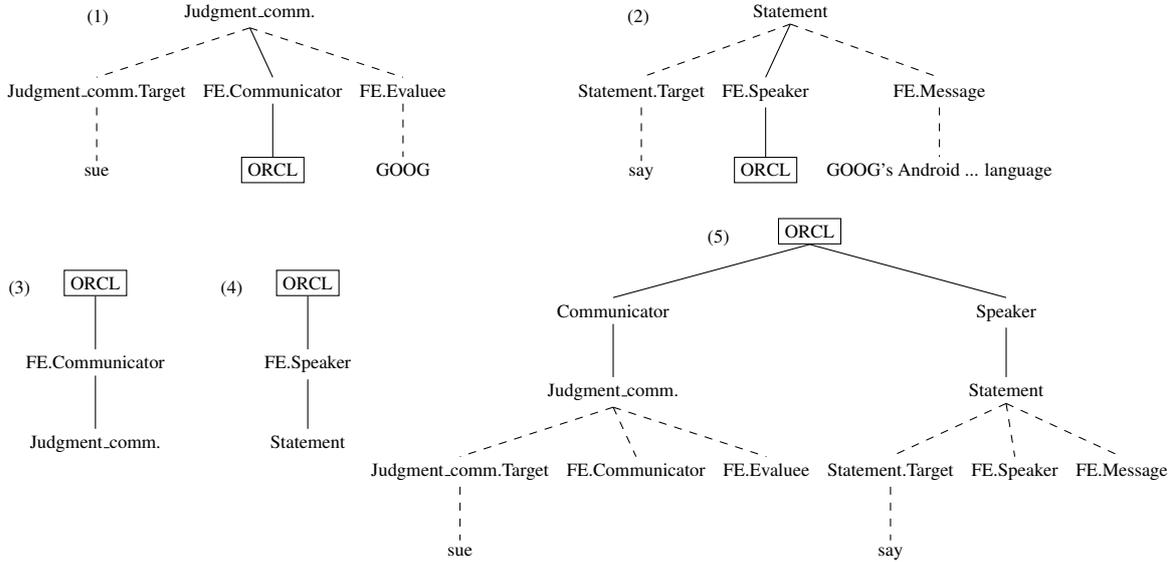


Figure 2: Constructing a tree representation for the designated object *Oracle* in sentence shown.

and  $h_i(T_2) = \sum_{n_2 \in N_{T_2}} I_i(n_2)$ . Their similarity can be efficiently computed by the inner product,

$$\begin{aligned}
K(T_1, T_2) &= \mathbf{h}(T_1) \cdot \mathbf{h}(T_2) \\
&= \sum_i h_i(T_1) h_i(T_2) \\
&= \sum_i (\sum_{n_1 \in N_{T_1}} I_i(n_1)) (\sum_{n_2 \in N_{T_2}} I_i(n_2)) \\
&= \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \sum_i I_i(n_1) I_i(n_2) \\
&= \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)
\end{aligned}$$

where  $\Delta(n_1, n_2)$  is the number of common fragments rooted in the nodes  $n_1$  and  $n_2$ . If the productions of these two nodes (themselves and their immediate children) differ,  $\Delta(n_1, n_2) = 0$ ; otherwise iterate their children recursively to evaluate  $\Delta(n_1, n_2) = \prod_j^{|\text{children}|} (\sigma + \Delta(c_{n_1}^j, c_{n_2}^j))$ , where  $\sigma = 0$  for ST kernel and  $\sigma = 1$  for SST kernel.

The kernel computational complexity is  $O(|N_{T_1}| \times |N_{T_2}|)$ , where all pairwise comparisons are carried out between  $T_1$  and  $T_2$ . However, there are fast algorithms for kernel computation that run in linear time on average, either by dynamic programming (Collins and Duffy, 2002), or pre-sorting production rules before training (Moschitti, 2006). We use the latter.

## 5 Dataset

We use publicly available financial news from Reuters from January 2007 through August 2012. This time frame includes a severe economic downturn in 2007-2010 followed by a modest recovery in 2011-2012.

An information extraction pipeline is used to pre-process the data. News full text is extracted from HTML. The timestamp of the news is extracted for a later alignment with stock price information, which will be discussed in section 6. The company mentioned is identified by a rule-based matching of a finite list of companies.

There are a total of 10 sectors in the Global Industry Classification Standard (GICS), an industry taxonomy used by the S&P 500.<sup>2</sup> To explore our approach for this domain, we select three sectors for our experiment: Telecommunication Services (TS, the sector with the smallest number of companies), Information Technology (IT), and Consumer Staples (CS), due to our familiarity with the companies in these sectors and an expectation of different characteristics they may exhibit. In the expectation there would be semantic differences associated with these sectors, experiments are performed independently for each sector. There are also differences in the number of companies in the sector, and the amount of news.

We bin news articles by sector. We remove articles that only list stock prices or only show tables of accounting reports. The first preprocessing step is to extract sentences that mention the

<sup>2</sup>Standard & Poor’s 500 is an equity market index that includes 500 U.S. leading companies in leading industries.

	CS (N=40)	IT (N=69)	TS (N=8)
avg # news	5,702±749	13446±1,272	2,177±188
avg # sentences	16,090±2,316	48,929±5,927	6,970±1,383
avg # com./sent.	1.07±0.01	1.06±0.20	1.14±0.03
avg # total	17,131±2,339	51,306±8,637	7,947±1,576

Table 3: Data statistics of mean and standard deviation by year from January 2007 to August 2012, for three sectors, with the number of companies.

relevant companies. Each data instance is a sentence and one of the target companies it mentions. Table 3 summarizes the data statistics. For example, the consumer staples sector has 40 companies. It has an average of 5,702 news articles (16,090 sentences) per year. Each sentence that mentions a consumer staple company mentions 1.07 companies on average. On average, this sector has 17,131 instances per year.

## 6 Experiments

Our current experiments are carried out for each year, training on one year and testing on the next. The choice to use a coarse time interval with no overlap was an expedience to permit more numerous exploratory experiments, given the computational resources these experiments require. We test the influence of news to predict (1) a change in stock price (*change* task), and (2) the polarity of change (increase vs. decrease; *polarity* task). Experiments evaluate the FWD and SemTree feature spaces compared to two baselines: bag-of-words (BOW) and supervised latent Dirichlet allocation (sLDA) (Blei and McAuliffe, 2007). BOW includes features of unigram, bigram and trigram. sLDA is a statistical model to classify documents based on LDA topic models, using labeled data. It has been applied to and shown good performance in topical text classification, collaborative filtering, and web page popularity prediction problems.

### 6.1 Labels, Evaluation Metrics, and Settings

We align publicly available daily stock price data from Yahoo Finance with the Reuters news using a method to avoid back-casting. In particular, we use the daily adjusted closing price - the price quoted at the end of a trading day (4PM US Eastern Time), then adjusted by dividends, stock split, and other corporate actions. We create two types of labels for news documents using the price data, to label the existence of a change and the direction of change. Both tasks are treated as binary classification problems. Based on the finding of

a one-day delay of the price response to the information embedded in the news by Tetlock et al. (2008), we use  $\Delta t = 1$  in our experiment. To constrain the number of parameters, we also use a threshold value ( $r$ ) of a 2% change, based on the distribution of price changes across our data. In future work, this could be tuned to sector or time.

$$\text{change} = \begin{cases} +1 & \text{if } \frac{|p_{t(0)+\Delta t} - p_{t(-1)}|}{p_{t(-1)}} > r \\ -1 & \text{otherwise} \end{cases}$$

$$\text{polarity} = \begin{cases} +1 & \text{if } p_{t(0)+\Delta t} > p_{t(-1)} \text{ and } \text{change} = +1 \\ -1 & \text{if } p_{t(0)+\Delta t} < p_{t(-1)} \text{ and } \text{change} = +1 \end{cases}$$

$p_{t(-1)}$  is the adjusted closing price at the end of the last trading day, and  $p_{t(0)+\Delta t}$  is the price of the end of the trading day after the  $\Delta t$  day delay. Only the instances with changes are included in the *polarity* task.

There is high variance across years in the proportion of positive labels, and often highly skewed classes in one direction or the other. The average ratios of +/- classes for *change* and *polarity* over the six years' data are 0.73 (std=0.35) and 1.12 (std=0.25), respectively. Because the time frame for our experiments includes an economic crisis followed by a recovery period, we note that the ratio between increase and decrease of price flips between 2007, where it is 1.40, and 2008, where it is 0.71. Accuracy is very sensitive to skew: when a class has low frequency, accuracy can be high using a baseline that makes prediction on the majority class. Given the high data skew, and the large changes from year to year in positive versus negative skew, we use a more robust evaluation metric.

Our evaluation relies on the Matthews correlation coefficient (MCC, also known as the  $\phi$ -coefficient) (Matthews, 1975) to avoid the bias of accuracy due to data skew, and to produce a robust summary score independent of whether the positive class is skewed to the majority or minority. In contrast to f-measure, which is a class-specific weighted average of precision and recall, and whose weighted version depends on a choice of whether the class-specific weights should come from the training or testing data, MCC is a single summary value that incorporates all 4 cells of a  $2 \times 2$  confusion matrix (TP, FP, TN and FN for True or False Positive or Negative). We have also observed that MCC has a lower relative standard deviation than f-measure.

For a  $2 \times 2$  contingency table, MCC corresponds to the square root of the average  $\chi^2$  statistic  $\sqrt{\chi^2/n}$ , with values in  $[-1,1]$ . It has been sug-

Change				
test years	BOW	sLDA	FWD	SemTreeFWD
Consumer Staples				
2008-2010	0.1015	0.0774	0.1079	0.1426
2011-2012	0.1663	0.1203	0.1664	0.1736
5 years	0.1274	0.0945	0.1313	0.1550
Information Technology				
2008-2010	0.0580	0.0585	0.0701	0.0846
2011-2012	0.0894	0.0681	0.1076	0.1273
5 years	0.0705	0.0623	0.0851	0.1017
Telecommunication Services				
2008-2010	0.1501	0.1615	0.1497	0.2409
2011-2012	0.2256	0.2084	0.2191	0.4009
5 years	0.1803	0.1803	0.1774	0.3049
Polarity				
Consumer Staples				
2008-2010	0.0359	0.0383	0.0956	0.1054
2011-2012	0.0938	0.0270	0.1131	0.1285
5 years	0.0590	0.0338	0.1026	0.1147
p-value		>>0.1000	0.0918	<b>0.0489</b>
Information Technology				
2008-2010	0.0551	0.0332	0.0697	0.0763
2011-2012	0.0591	0.0516	0.0764	0.0857
5 years	0.0567	0.0405	0.0723	0.0801
p-value		0.0626	0.0948	<b>0.0103</b>
Telecommunication Services				
2008-2010	0.0402	0.0464	0.0821	0.0745
2011-2012	0.0366	0.0781	0.0611	0.0809
5 years	0.0388	0.0591	0.0737	0.0770
p-value		>>0.1000	0.0950	<b>0.0222</b>

Table 4: Average MCC for the change and polarity tasks by feature representation, for 2008-2010; for 2011-2012; for all 5 years and associated p-values of ANOVAs for comparison to BOW.

gested as one of the best methods to summarize into a single value the confusion matrix of a binary classification task (Jurman and Furlanello, 2010; Baldi et al., 2000). Given the confusion matrix  $\begin{pmatrix} TP & FN \\ FP & TN \end{pmatrix}$ :

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}.$$

All sentences with at least one company mention are used for the experiment. We remove stop words and use Stanford CoreNLP for part-of-speech tagging and named entity recognition. Models are constructed using linear kernel support vector machines for both classification tasks. SVM-light with tree kernels<sup>3</sup> (Joachims, 2006; Moschitti, 2006) is used for both the FWD and SemTree feature spaces.

## 6.2 Results

Table 4 shows the mean MCC values for each task, for each sector. Separate means are shown for the test years of financial crisis (2008-2010) and economic recovery (2011-2012) to highlight the differences in performance that might result from market volatility.

<sup>3</sup>SVM-light: <http://svmlight.joachims.org> and Tree Kernels in SVM-light: <http://disi.unitn.it/moschitti/Tree-Kernel.htm>.

pos. 1	dow, investors, index, retail, data
pos. 2	costs, food, price, prices, named_entity_4
neu. 1	q3, q1, nov, q2, apr
neu. 2	cents, million, share, year, quarter
neg. 1	cut, sales, prices, hurt, disappointing
neg. 2	percent, call, company, fell, named_entity_7

Table 5: Sample sLDA topics for consumer staples for test year 2010 (train on 2009), polarity task.

SemTree combined with FWD (SemTreeFWD) generally gives the best performance in both *change* and *polarity* tasks. SemTree results here are based on the subset tree (SST) kernel, because of its greater precision in computing common frame structures and consistently better performance over the subtree (ST) kernel. SemTree also provides interpretable features for manual analysis as discussed in the next section.

Analysis of Variance (ANOVA) tests were performed on the full 5 years for each sector, to compare each feature representation as a predictor of MCC score with the baseline BOW. The ANOVAs yield the p-values shown in Table 4. There were no significant differences from BOW on the *change* task. For *polarity* detection, SemTreeFWD was significantly better than BOW for each sector (see boldface p-values). No other method was significantly better than BOW, although FWD approaches significance on all sectors, and sLDA approaches significance on IT.

sLDA has promising MCC scores for the telecommunication sector, which has only 8 companies, thus many fewer data instances. Table 5 displays a sample of sLDA topics with good performance on polarity for the consumer staples sector for training year 2009. The positive topics are related to stock index details and retail data. The negative topics contain many words with negative sentiment (e.g., *hurt*, *disappointing*).

## 7 Discussion

### 7.1 Semantic Parse Quality

In general, SEMAFOR parses capture most of the important frames for our purposes. There is, however, significant room for improvement. On a small, randomly selected sample of sentences from all three sectors, two of the authors working independently evaluated the semantic parses, with approximately 80% agreement. Some of the inaccuracies in frame parses result from errors prior to the SEMAFOR parse, such as tokenization or

```

+ (Target(jump))
+ (RECIPIENT(Receiving))
+ (VICTIM(Defend))
+ (PERCEIVER_AGENTIVE(Perception_active(Target)
(PERCEIVER_AGENTIVE(PHENOMENON)))
+ (DONOR(Giving(Target)(THEME)(DONOR)))
+ (Target(beats))
...
- (PHENOMENON(Perception_active(Target)(PERCEIVER
_AGENTIVE)(PHENOMENON)))
- (TRIGGER(Response))
- (Target(cuts))
- (VICTIM(Cause_Harm(Target(hurt))(VICTIM)))

```

Figure 3: Best performing SemTree fragments for increase (+) and decrease (-) of price for consumer staples sector across training years.

dependency parsing errors. The average sentence length for the sample was 33.3 words, with an average of 14 frames per sentence, 3 of them with a GICS company as a role filler. Because SemTree encodes only the frames containing a designated object (company), these are the frames we evaluated. On average, about half the frames with a designated object were correct, and two thirds of those frames we judged to be important. Besides errors due to incorrect tokenization or dependency parsing, we observed that about 8% to 10% of frames were incorrectly assigned to due word sense ambiguity.

## 7.2 Feature Analysis

The experimental results show the SemTree space to be the one representation tested here that is significantly better than BOW, but only for the *polarity* task. Post hoc analysis indicates this may be due to the aptness of semantic frame parsing for polarity. Limitations in our treatment of time point to directions for improvement regarding the *change* task.

Some strengths of our approach are the separate treatment of different sectors, and the benefits of SemTree features. To analyze which were the best performing features within sectors, we extracted the best performing frame fragments for the *polarity* task using a tree kernel feature engineering method presented in Pighin and Moschitti (2009). The algorithm selects the most relevant features in accordance with the weights estimated by SVM, and uses these features to build an explicit representation of the kernel space. Figure 3 shows the best performing SemTree fragments of the *polarity* task for the consumer staples sector.

Recall that we hypothesized differences in

semantic frame features across sectors. This shows up as large differences in the strength of features across sectors. More strikingly, the same feature can differ in polarity across sectors. For example, in consumer staples, (EVAL-UEE(*Judgment\_communication*)) has positive polarity, compared with negative polarity in information technology sector. The examples we see indicate that the positive cases pertain to aggressive retail practices that lead to lawsuits with only small fines, but whose larger impact benefits the bottom line. A typical case is the sentence, *The plaintiffs accused Wal-Mart of discriminating against disabled customers by mounting “point-of-sale” terminals in many stores at elevated heights that cannot be reached.* Lawsuits in the IT sector, on the other hand, are often about technology patent disputes, and are more negative, as illustrated by our example sentence in Figure 2.

SemTree features capture the differences between semantic roles for the same frame, and between the same semantic role in different frames. For example, the PERCEIVER\_AGENTIVE role of the *Perception\_active* frame contributes to prediction of an increase in price, as in *R.J. Reynolds is watching this situation closely and will respond as appropriate.* Conversely, a company that fills the PHENOMENON role of the same frame contributes to prediction of a price decrease, as in *Investors will get a clearer look at how the market values the Philip Morris tobacco businesses when Altria Group Inc. “when-issued” shares begin trading on Tuesday.* When a company fills the VICTIM role in the *Cause\_harm* frame, this can predict a decrease in price, as in *Hershey has been hurt by soaring prices for cocoa, energy and other commodities,* whereas filling the VICTIM role in the *Defend* frame is associated with an increase in price, as in *At Berkshire’s annual shareholder meeting earlier this month, Warren Buffett defended Wal-Mart, saying the scandal did not change his opinion of the company.*

One weakness of our approach that we discussed above is that there is a strong effect of time that we do not address. The same SemTree feature can be predictive for one time period and not for another. (GOODS(*Commerce\_sell*)) is related to a decrease in price for 2008 and 2009 but to an increase in price for 2010-2012. There is clearly an influence of the overall economic context that we do not take into account. For example,

the practices of acquiring or selling a business are different in downturning versus recovering markets. An important observation of the MCC values, especially in the case of SemTreeFWD is that MCC increases during the years 2011-2012. We attribute this change to the difficulty of predicting stock price trends when there is the high volatility typical of a financial crisis. The effect of news on volatility, however, can be explored independently. For example, Creamer et al. (2012) detect a strong association.

Another weakness of our approach is that we take sentences out of context, which can lead to prediction errors. For example, the sentence *Longs' real estate assets alone are worth some \$2.9 billion, or \$71.50 per share, Ackman wrote, meaning that [CVS] would essentially be paying for real estate, but gaining Longs' pharmacy benefit management business and retail operations for free* is treated as predicting a positive polarity for CVS. This would be accurate if CVS was actually going to acquire Longs' business. Later in the same news item, however, there is a sentence indicating that the sale will not go through, which predicts negative polarity for CVS: *Pershing Square Capital Management said on Thursday it won't support a tender offer from CVS Caremark Corp for rival Longs Drug Stores Corp because the offer price "materially understates the fair value of the company," according to a filing.*

## 8 Conclusion

We have presented a model for predicting stock price movement from news. We proposed FWD (Frames, BOW, and part-of-speech specific DAL) features and SemTree data representations. Our semantic frame-based model benefits from tree kernel learning using support vector machines. The experimental results for our feature representation perform significantly better than BOW on the *polarity* task, and show promise on the *change* task. It also facilitates human interpretable analysis to understand the relation between a company's market value and its business activities. The signals generated by this algorithm could improve the prediction of a financial time series model, such as ADS (Rydberg and Shephard, 2003).

Our future work will consider the contextual information for sentence selection, and an aggregation of weighted news content based on the decay effect over time for individual companies. We plan

to use a moving window for training and testing. We will also explore different labeling methods, such as a threshold for price change tuned by sectors and background economics.

## 9 Acknowledgements

The authors thank the anonymous reviewers for their insightful comments.

## References

- Apoorv Agarwal, Fadi Biadisy, and Kathleen Mckeown. 2009. Contextual phrase-level polarity analysis using lexical affect scoring and syntactic N-grams. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 24–32, Athens, Greece, March. Association for Computational Linguistics.
- Apoorv Agarwal, Boyi Xie, Iliia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media, LSM '11*, pages 30–38. Association for Computational Linguistics.
- Pierre Baldi, Søren Brunak, Yves Chauvin, Claus A. F. Andersen, and Henrik Nielsen. 2000. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16:412 – 424.
- Roy Bar-Haim, Elad Dinur, Ronen Feldman, Moshe Fresko, and Guy Goldstein. 2011. Identifying and following expert investors in stock microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1310–1319, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- David M. Blei and Jon D. McAuliffe. 2007. Supervised topic models. In *Advances in Neural Information Processing Systems, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6*.
- Christopher Chua, Maria Milosavljevic, and James R. Curran. 2009. A sentiment detection engine for internet stock message boards. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, pages 89–93, Sydney, Australia, December.
- Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

- Germán G. Creamer, Yong Ren, and Jeffrey V. Nickerson. 2012. A Longitudinal Analysis of Asset Return, Volatility and Corporate News Network. In *Business Intelligence Congress 3 Proceedings*.
- Dipanjan Das and Noah A. Smith. 2011. Semi-supervised frame-semantic parsing for unknown predicates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1435–1444, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dipanjan Das and Noah A. Smith. 2012. Graph-based lexicon expansion with sparsity-inducing penalties. In *HLT-NAACL*, pages 677–687. The Association for Computational Linguistics.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*.
- Ann Devitt and Khurshid Ahmad. 2007. Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 984–991, Prague, Czech Republic, June. Association for Computational Linguistics.
- William Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. *Proceedings of the 20th International Conference on Computational Linguistics*.
- Joseph Engelberg and Christopher A. Parsons. 2011. The causal impact of media in financial markets. *Journal of Finance*, 66(1):67–97.
- Charles J. Fillmore, Christopher R. Johnson, and Miriam R. L. Petruck. 2003. Background to Framenet. *International Journal of Lexicography*, 16(3):235–250, September.
- Charles J. Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences*, 280(1):20–32.
- Syed Aqueel Haider and Rishabh Mehrotra. 2011. Corporate news classification and valence prediction: A supervised approach. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*, pages 175–181, Portland, Oregon, June. Association for Computational Linguistics.
- Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 217–226, New York, NY, USA. ACM.
- Giuseppe Jurman and Cesare Furlanello. 2010. A unifying view for performance measures in multi-class prediction. *ArXiv e-prints*.
- Soo-Min Kim and Eduard Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, SST '06, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Shimon Kogan, Dimitry Levin, Bryan R. Routledge, Jacob S. Sagi, and Noah A. Smith. 2009. Predicting risk from financial reports with regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 272–280, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Victor Lavrenko, Matt Schmill, Dawn Lawrie, Paul Ogilvie, David Jensen, and James Allan. 2000. Mining of concurrent text and time series. In *Proceedings of the 6th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining Workshop on Text Mining*, pages 37–44.
- Ronny Luss and Alexandre d'Aspremont. 2008. Predicting abnormal returns from news using text classification. *CoRR*, abs/0809.2792.
- Brian W. Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442 – 451.
- Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*.
- Daniele Pighin and Alessandro Moschitti. 2009. Reverse engineering of tree kernel feature spaces. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore*, pages 111–120.
- Eduardo J. Ruiz, Vagelis Hristidis, Carlos Castillo, Aristides Gionis, and Alejandro Jaimes. 2012. Correlating financial time series with micro-blogging activity. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 513–522, New York, NY, USA. ACM.
- Josef Ruppenhofer and Ines Rehbein. 2012. Semantic frames as an anchor representation for sentiment analysis. In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '12, pages 104–109, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Tina H. Rydberg and Neil Shephard. 2003. Dynamics of Trade-by-Trade Price Movements: Decomposition and Models. *Journal of Financial Econometrics*, 1(1):2–25.

Paul C. Tetlock, Maytal Saar-Tsechansky, and Sofus Macskassy. 2008. More than Words: Quantifying Language to Measure Firms' Fundamentals. *The Journal of Finance*.

Paul C. Tetlock. 2007. Giving Content to Investor Sentiment: The Role of Media in the Stock Market. *The Journal of Finance*.

Cynthia M. Whissel. 1989. The dictionary of affect in language. *Emotion: Theory, Research, and Experience*, 39(4):113–131.

# Density Maximization in Context-Sense Metric Space for All-words WSD

Koichi Tanigaki<sup>†‡</sup> Mitsuteru Shiba<sup>†</sup> Tatsuji Munaka<sup>†</sup> Yoshinori Sagisaka<sup>‡</sup>

<sup>†</sup> Information Technology R&D Center, Mitsubishi Electric Corporation  
5-1-1 Ofuna, Kamakura, Kanagawa 247-8501, Japan

<sup>‡</sup> Global Information and Telecommunication Institute, Waseda University  
1-3-10 Nishi-Waseda, Shinjuku-ku, Tokyo 169-0051, Japan

## Abstract

This paper proposes a novel *smoothing model* with a *combinatorial optimization scheme* for all-words word sense disambiguation from untagged corpora. By generalizing discrete senses to a continuum, we introduce a smoothing in context-sense space to cope with *data-sparsity* resulting from a large variety of linguistic context and sense, as well as to exploit *sense-interdependency* among the words in the same text string. Through the smoothing, all the optimal senses are obtained at one time under maximum marginal likelihood criterion, by competitive probabilistic kernels made to reinforce one another among nearby words, and to suppress conflicting sense hypotheses within the same word. Experimental results confirmed the superiority of the proposed method over conventional ones by showing the better performances beyond most-frequent-sense baseline performance where none of SemEval-2 unsupervised systems reached.

## 1 Introduction

Word Sense Disambiguation (WSD) is a task to identify the intended sense of a word based on its context. *All-words WSD* is its variant, where all the unrestricted running words in text are expected to be disambiguated. In the all-words task, all the senses in a dictionary are potentially the target destination of classification, and purely supervised approaches inherently suffer from *data-sparsity* problem. The all-words task is also characterized by *sense-interdependency* of target words. As the target words are typically taken from the same

text string, they are naturally expected to be inter-related. Disambiguation of a word should affect other words as an important clue.

From such characteristics of the task, knowledge-based unsupervised approaches have been extensively studied. They compute dictionary-based sense similarity to find the most related senses among the words within a certain range of text. (For reviews, see (Agirre and Edmonds, 2006; Navigli, 2009).) In recent years, graph-based methods have attracted considerable attentions (Mihalcea, 2005; Navigli and Lapata, 2007; Agirre and Soroa, 2009). On the graph structure of lexical knowledge base (LKB), random-walk or other well-known graph-based techniques have been applied to find mutually related senses among target words. Unlike earlier studies disambiguating word-by-word, the graph-based methods obtain *sense-interdependent* solution for target words. However, those methods mainly focus on modeling sense distribution and have less attention to contextual smoothing/generalization beyond immediate context.

There exist several studies that enrich immediate context with large corpus statistics. McCarthy et al. (2004) proposed a method to combine sense similarity with distributional similarity and configured predominant sense score. Distributional similarity was used to weight the influence of context words, based on large-scale statistics. The method achieved successful WSD accuracy. Agirre et al. (2009) used a *k*-nearest words on distributional similarity as context words. They apply a LKB graph-based WSD to a target word together with the distributional context words, and showed that it yields better results on a domain dataset than just using immediate context words. Though these

studies are word-by-word WSD for target words, they demonstrated the effectiveness to enrich immediate context by corpus statistics.

This paper proposes a *smoothing model* that integrates dictionary-based semantic similarity and corpus-based context statistics, where a *combinatorial optimization scheme* is employed to deal with sense interdependency of the all-words WSD task. The rest of this paper is structured as follows. We first describe our smoothing model in the following section. The combinatorial optimization method with the model is described in Section 3. Section 4 describes a specific implementation for evaluation. The evaluation is performed with the SemEval-2 English all-words dataset. We present the performance in Section 5. In Section 6 we discuss whether the intended context-to-sense mapping and the sense-interdependency are properly modeled. Finally we review related studies in Section 7 and conclude in Section 8.

## 2 Smoothing Model

Let us introduce in this section the basic idea for modeling context-to-sense mapping. The distance (or similarity) metrics are assumed to be given for context and for sense. A specific implementation of these metrics is described later in this paper, for now the context metric is generalized with a distance function  $d_x(\cdot, \cdot)$  and the sense metric with  $d_s(\cdot, \cdot)$ . Actually these functions may be arbitrary ones that accept two elements and return a positive real number.

Now suppose we are given a dataset concerning  $N$  number of target words. This dataset is denoted by  $X = \{x_i\}_{i=1}^N$ , where  $x_i$  corresponds to the context of the  $i$ -th word but not the word by itself. For each  $x_i$ , the intended sense of the word is to be found in a set of sense candidates  $S_i = \{s_{ij}\}_{j=1}^{M_i} \subseteq S$ , where  $M_i$  is the number of sense candidates for the  $i$ -th word,  $S$  is the whole set of sense inventories in a dictionary. Let the two-tuple  $h_{ij} = (x_i, s_{ij})$  be the hypothesis that the intended sense in  $x_i$  is  $s_{ij}$ . The hypothesis is an element of the direct product  $H = X \times S$ . As  $(X, d_x)$  and  $(S, d_s)$  each composes a metric space,  $H$  is also a metric space, provided a proper distance definition with  $d_x$  and  $d_s$ .

Here, we treat the space  $H$  as a continuous one, which means that we assume *the relationship between context and sense can be generalized in continuous fashion*. In natural language processing,

continuity has been sometimes assumed for linguistic phenomena including word context for corpus based WSD. As for classes or senses, it may not be a common assumption. However, when the classes for all-words WSD are enormous, fine-grained, and can be associated with distance, we can rather naturally assume the continuity also for senses. According to the nature of continuity, once given a hypothesis  $h_{ij}$  for a certain word, we can extrapolate the hypothesis for another word of another sense  $h_{i'j'} = (x_{i'}, s_{i'j'})$  sufficiently close to  $h_{ij}$ . Using a Gaussian kernel (Parzen, 1962) as a smoothing model, the probability density extrapolated at  $h_{i'j'}$  given  $h_{ij}$  is defined by their distance as follows:

$$\begin{aligned} \mathcal{K}(h_{ij}, h_{i'j'}) & \\ \equiv \frac{1}{2\pi\sigma_x\sigma_s} \exp \left[ -\frac{d_x^2(x_i, x_{i'})}{2\sigma_x^2} - \frac{d_s^2(s_{ij}, s_{i'j'})}{2\sigma_s^2} \right], & \end{aligned} \quad (1)$$

where  $\sigma_x$  and  $\sigma_s$  are parameters of positive real number  $\sigma_x, \sigma_s \in \mathbb{R}^+$  called *kernel bandwidths*. They control the smoothing intensity in context and in sense, respectively.

Our objective is to determine the optimal sense for all the target words simultaneously. It is essentially a 0-1 integer programming problem, and is not computationally tractable. We relax the integer constraints by introducing a *sense probability* parameter  $\pi_{ij}$  corresponding to each  $h_{ij}$ .  $\pi_{ij}$  denotes the probability by which  $h_{ij}$  is true. As  $\pi_{ij}$  is a probability, it satisfies the constraints  $\forall i \sum_j \pi_{ij} = 1$  and  $\forall i, j \ 0 \leq \pi_{ij} \leq 1$ . The probability density extrapolated at  $h_{i'j'}$  by a *probabilistic* hypothesis  $h_{ij}$  is given as follows:

$$\mathcal{Q}_{ij}(h_{i'j'}) \propto \pi_{ij} \mathcal{K}(h_{ij}, h_{i'j'}). \quad (2)$$

The proposed model is illustrated in Figure 1. Due to the limitation of drawing, both the context metric space and the sense metric space are drawn schematically as 1-dimensional spaces (axes), actually arbitrary metric spaces similarity-based or feature-based are applicable. The product metric space of the context metric space and the sense metric space composes a hypothesis space. In the hypothesis space,  $n$  sense hypotheses for a certain word is represented as  $n$  points on the hyperplane that spreads across the sense metric space. The two small circles in the middle of the figure represent the two sense hypotheses for a single word. The position of a hypothesis represents which sense is assigned to the current word in

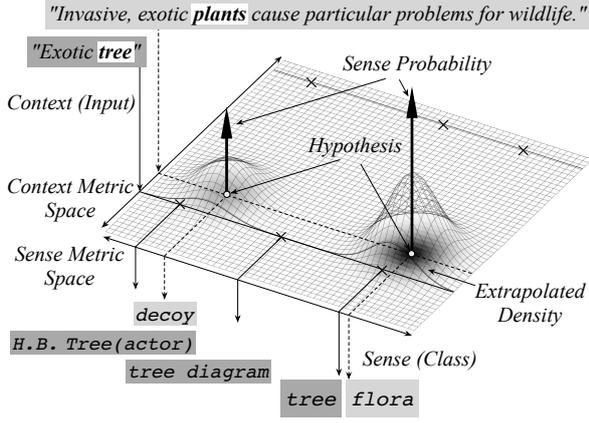


Figure 1: Proposed probability distribution model for context-to-sense mapping space.

what context. The upward arrow on a hypothesis represents the magnitude of its probability.

Centered on each hypotheses, a Gaussian kernel is placed as a smoothing model. It extrapolates the hypotheses of other words around it. In accordance with geometric intuition, intensity of extrapolation is affected by the distance from a hypothesis, and by the probability of the hypothesis by itself. Extrapolated probability density is represented by shadow thickness and surface height. If there is another word in nearby context, the kernels can validate the sense of that word. In the figure, there are two kernels in the context “*Invasive, exotic ...*”. They are two competing hypothesis for the senses *decoy* and *flora* of the word *plants*. These kernels affect the senses of another ambiguous word *tree* in nearby context “*Exotic ...*”, and extrapolate the most at the sense *tree* nearby *flora*. The extrapolation has non-linear effect. It affects little to the word far away in context or in sense as is the case for the background word in the figure. Strength of smoothing is determined by kernel bandwidths. Wider bandwidths bring stronger effect of generalization to further hypotheses, but too wide bandwidths smooth out detailed structure. The bandwidths are the key for disambiguation, therefore they are to be optimized on a dataset together with sense probabilities.

### 3 Simultaneous Optimization of All-words WSD

Given the smoothing model to extrapolate the senses of other words, we now make its instances interact to obtain the optimal combination of senses for all the words.

### 3.1 Likelihood Definition

Let us first define the likelihood of model parameters for a given dataset. The parameters consist of a context bandwidth  $\sigma_x$ , a sense bandwidth  $\sigma_s$ , and sense probabilities  $\pi_{ij}$  for all  $i$  and  $j$ . For convenience of description, the sense probabilities are all together denoted as a vector  $\boldsymbol{\pi} = (\dots, \pi_{ij}, \dots)^\top$ , in which actual order is not the matter.

Now remind that our dataset  $X = \{x_i\}_{i=1}^N$  is composed of  $N$  instances of *unlabeled* word context. We consider all the mappings from context to sense are latent, and find the optimal parameters by maximizing marginal pseudo likelihood based on probability density. The likelihood is defined as follows:

$$\mathcal{L}(\boldsymbol{\pi}, \sigma_x, \sigma_s; X) \equiv \ln \prod_i \sum_j \pi_{ij} \mathcal{Q}(h_{ij}), \quad (3)$$

where  $\prod_i$  denotes the product over  $x_i \in X$ ,  $\sum_j$  denotes the summation over all possible senses  $s_{ij} \in S_i$  for the current  $i$ -th context.  $\mathcal{Q}(h_{ij})$  denotes the probability density at  $h_{ij}$ . We compute  $\mathcal{Q}(h_{ij})$  using *leave-one-out cross-validation* (LOOCV), so as to prevent kernels from overfitting to themselves, as follows:

$$\begin{aligned} \mathcal{Q}(h_{ij}) & \\ \equiv \frac{1}{N - N_i} \sum_{i': w_{i'} \neq w_i} \sum_{j'} \pi_{i'j'} \mathcal{K}(h_{ij}, h_{i'j'}), & \quad (4) \end{aligned}$$

where  $N_i$  denotes the number of occurrences of a word type  $w_i$  in  $X$ , and  $\sum_{i': w_{i'} \neq w_i}$  denotes the summation over  $x_{i'} \in X$  except the case that the word type  $w_{i'}$  equals to  $w_i$ .  $\sum_{j'}$  denotes the summation over  $s_{i'j'} \in S_{i'}$ . We take as the unit of LOOCV not a word instance but a word type, because the instances of the same word type invariably have the same sense candidates, which still cause overfitting when optimizing the sense bandwidth.

### 3.2 Parameter Optimization

We are now ready to calculate the optimal senses. The optimal parameters  $\boldsymbol{\pi}^*$ ,  $\sigma_x^*$ ,  $\sigma_s^*$  are obtained by maximizing the likelihood  $\mathcal{L}$  subject to the constraints on  $\boldsymbol{\pi}$ , that is  $\forall i \sum_j \pi_{ij} = 1$ <sup>1</sup>. Using the Lagrange multipliers  $\{\lambda_i\}_{i=1}^N$  for every  $i$ -th constraint, the solution for the constrained maximiza-

<sup>1</sup>It is guaranteed that the other constraints  $\forall i, j 0 \leq \pi_{ij} \leq 1$  are satisfied according to Equation (7).

tion of  $\mathcal{L}$  is obtained as the solution for the equivalent unconstrained maximization of  $\check{\mathcal{L}}$  as follows:

$$\boldsymbol{\pi}^*, \sigma_x^*, \sigma_s^* = \arg \max_{\boldsymbol{\pi}, \sigma_x, \sigma_s} \check{\mathcal{L}}, \quad (5)$$

where

$$\check{\mathcal{L}} \equiv \mathcal{L} + \sum_i \lambda_i \left( \sum_j \pi_{ij} - 1 \right). \quad (6)$$

When we optimize the parameters, the first term of Equation (6) in the right-hand side acts *to reinforce* nearby hypotheses among different words, whereas the second term acts *to suppress* conflicting hypotheses of the same word.

Taking  $\nabla \check{\mathcal{L}} = 0$ , erasing  $\lambda_i$ , and rearranging, we obtain the optimal parameters as follows:

$$\pi_{ij} = \frac{\sum_{\substack{i', j' \\ w_{i'} \neq w_i}} \mathcal{R}_{i'j'}^{ij} + \sum_{\substack{i', j' \\ w_{i'} \neq w_i}} \mathcal{R}_{ij}^{i'j'}}{1 + \sum_j \sum_{\substack{i', j' \\ w_{i'} \neq w_i}} \mathcal{R}_{ij}^{i'j'}} \quad (7)$$

$$\sigma_x^2 = \frac{1}{N} \sum_{\substack{i, i', j, j' \\ w_{i'} \neq w_i}} \mathcal{R}_{i'j'}^{ij} d_x^2(x_i, x_{i'}) \quad (8)$$

$$\sigma_s^2 = \frac{1}{N} \sum_{\substack{i, i', j, j' \\ w_{i'} \neq w_i}} \mathcal{R}_{i'j'}^{ij} d_s^2(s_{ij}, s_{i'j'}), \quad (9)$$

where  $\mathcal{R}_{i'j'}^{ij}$  denotes the *responsibility* of  $h_{i'j'}$  to  $h_{ij}$ : the ratio of total expected density at  $h_{ij}$ , taken up by the expected density extrapolated by  $h_{i'j'}$ , normalized to the total for  $x_i$  be 1. It is defined as

$$\mathcal{R}_{i'j'}^{ij} \equiv \frac{\pi_{ij} \mathcal{Q}_{i'j'}(h_{ij})}{\sum_j \pi_{ij} \mathcal{Q}(h_{ij})}. \quad (10)$$

$\mathcal{Q}_{i'j'}(h_{ij})$  denotes the probability density at  $h_{ij}$  extrapolated by  $h_{i'j'}$  alone, defined as follows:

$$\mathcal{Q}_{i'j'}(h_{ij}) \equiv \frac{1}{N - N_i} \pi_{i'j'} \mathcal{K}(h_{ij}, h_{i'j'}). \quad (11)$$

Intuitively, Equations (7)-(9) are interpreted as follows. As for Equation (7), the right-hand side of the equation can be divided as the left term and the right term both in the numerator and in the denominator. The left term requires  $\pi_{ij}$  to agree with the ratio of responsibility *of the whole to  $h_{ij}$* . The right term requires  $\pi_{ij}$  to agree with the ratio of responsibility *of  $h_{ij}$  to the whole*. As for Equation (8), (9), the optimal solution is the mean squared distance in context, and in sense, weighted by responsibility.

To obtain the actual values of the optimal parameters, EM algorithm (Dempster et al., 1977) is applied. This is because Equations (7)-(9) are circular definitions, which include the objective parameters implicitly in the right hand side, thus the solution is not obtained analytically. EM algorithm is an iterative method for finding maximum likelihood estimates of parameters in statistical models, where the model depends on unobserved latent variables. Applying the EM algorithm to our model, we obtain the following steps:

**Step 1. Initialization:** Set initial values to  $\boldsymbol{\pi}$ ,  $\sigma_x$ , and  $\sigma_s$ . As for sense probabilities, we set the uniform probability in accordance with the number of sense candidates, thereby  $\pi_{ij} \leftarrow |S_i|^{-1}$ , where  $|S_i|$  denotes the size of  $S_i$ . As for bandwidths, we set the mean squared distance in each metric; thereby  $\sigma_x^2 \leftarrow N^{-1} \sum_{i, i'} d_x^2(x_i, x_{i'})$  for context bandwidth, and  $\sigma_s^2 \leftarrow (\sum_i |S_i|)^{-1} \sum_{i, i'} \sum_{j, j'} d_s^2(s_{ij}, s_{i'j'})$  for sense bandwidth.

**Step 2. Expectation:** Using the current parameters  $\boldsymbol{\pi}$ ,  $\sigma_x$ , and  $\sigma_s$ , calculate the responsibilities  $\mathcal{R}_{i'j'}^{ij}$  according to Equation (10).

**Step 3. Maximization:** Using the current responsibility  $\mathcal{R}_{i'j'}^{ij}$ , update the parameters  $\boldsymbol{\pi}$ ,  $\sigma_x$ , and  $\sigma_s$ , according to Equation (7)-(9).

**Step 4. Convergence test:** Compute the likelihood. If its ratio to the previous iteration is sufficiently small, or predetermined number of iterations has been reached, then terminate the iteration. Otherwise go back to Step 2.

To visualize how it works, we applied the above EM algorithm to pseudo 2-dimensional data. The results are shown in Figure 2. It simulates WSD for an  $N = 5$  words dataset, whose contexts are depicted by five lines. The sense hypotheses are depicted by twelve upward arrows. At the base of each arrow, there is a Gaussian kernel. Shadow thickness and surface height represents the composite probability distribution of all the twelve kernels. Through the iterative parameter update, sense probabilities and kernel bandwidths were optimized to the dataset. Figure 2(a) illustrates the initial status, where all the sense hypothesis are equivalently probable, thus they are in the most ambiguous status. Initial bandwidths are set to the mean squared distance of all the hypotheses pairs,

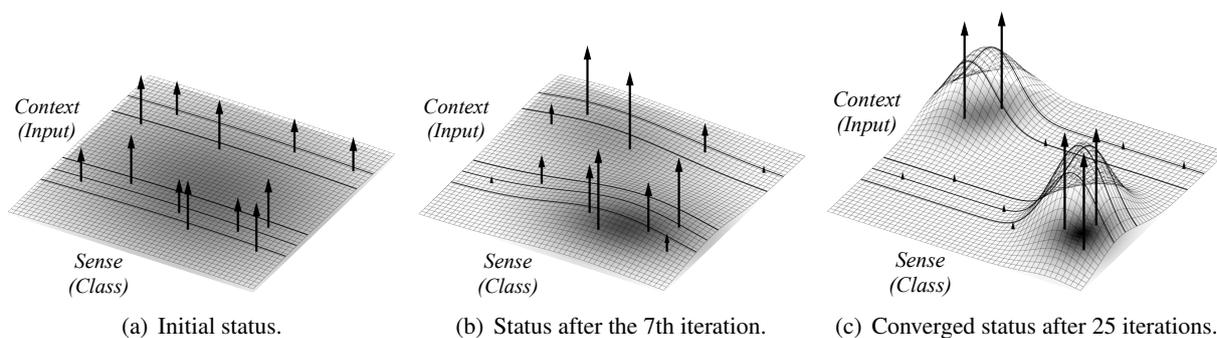


Figure 2: Pseudo 2D data simulation to visualize the dynamics of the proposed simultaneous all-words WSD with ambiguous five words and twelve sense hypotheses. (There are twelve Gaussian kernels at the base of each arrow, though the figure shows just their composite distribution. Those kernels reinforce and compete one another while being fitted their affecting range, and finally settle down to the most consistent interpretation for the words with appropriate generalization. For the dynamics with an actual dataset, see Figure 5.)

which is rather broad and makes kernels strongly smoothed, thus the model captures general structure of space. Figure 2(b) shows the status after the 7th iteration. Bandwidths are shrinking especially in context, and two context clusters, so to speak, two usages, are found. Figure 2(c) shows the status of convergence after 25 iterations. All the arrow lengths incline to either 1 or 0 along with their neighbors, thus all the five words are now disambiguated.

Note that this is not the conventional clustering of observed data. If, for instance, the Gaussian mixture clustering of 2-mixtures is applied to the positions of these hypotheses, it will find the clusters just like Figure 2(b) and will stop. The cluster centers are located at the means of hypotheses including miscellaneous alternatives not intended, thus the estimated probability distribution is, roughly speaking, offset toward the center of WordNet, which is not what we want. In contrast, the proposed method proceeds to Figure 2(c) and finds clusters *in the data after conflicting data is erased*. This is because our method is aiming at modeling not the disambiguation of cluster-memberships but the disambiguation of senses for each word.

#### 4 Metric Space Implementation

So far, we have dealt with general metrics for context and for sense. This section describes a specific implementation of those metrics employed in the evaluation. We followed the previous study by McCarthy et al. (2004), (2007), and implemented a type-based WSD. The context of word

instances are tied to the distributional context of the word type in a large corpus. To calculate sense similarities, we used the WordNet similarity package by Pedersen et al. (2004), version 2.05. Two measures proposed by Jiang and Conrath (1997) and Lesk (1986) were examined, which performed best in the previous study (McCarthy et al., 2004).

Distributional similarity (Lin, 1998) was computed among target words, based on the statistics of the test set and the background text provided as the official dataset of the SemEval-2 English all-words task (Agirre et al., 2010). Those texts were parsed using RASP parser (Briscoe et al., 2006) version 3.1, to obtain grammatical relations for the distributional similarity, as well as to obtain lemmata and part-of-speech (POS) tags which are required to look up the sense inventory of WordNet. Based on the distributional similarity, we just used  $k$ -nearest neighbor words as the context of each target word. Although it is an approximation, we can expect reliability improvement often seen by ignoring the lower part. In addition, this limitation of interactions highly reduces computational cost in particular when applying to larger-scale problems. To do this, the exhaustive sum  $\sum_{i, i': w_i \neq w_{i'}}$  in Equation (7)-(9) is altered by the local sum  $\sum_{i, i': (w_i, w_{i'}) \in P_{kNN}}$ , where  $P_{kNN}$  denotes the set of word pairs of which either is a  $k$ -nearest neighbors of the other. The normalizing factors 1,  $N$ , and  $N - N_i$  in Equation (7), (8)-(9), and (11) are altered by the actual sum of responsibilities within those neighbors as  $\sum_{i', j, j': (w_i, w_{i'}) \in P_{kNN}} \mathcal{R}_{i'j'}^{ij}$ ,

$\sum_{i, i', j, j': (w_i, w_{i'}) \in P_{k\text{NN}}} \mathcal{R}_{i'j'}^{ij}$ , and  
 $\sum_{i, i', j, j': (w_i, w_{i'}) \in P_{k\text{NN}} \wedge i \neq i'} \mathcal{R}_{i'j'}^{ij}$ , respectively.

To treat the above similarity functions of context and of sense as distance functions, we use the conversion:  $d(\cdot, \cdot) \equiv -\alpha \ln(f(\cdot, \cdot)/f_{\max})$ , where  $d$  denotes the objective distance function, i.e.,  $d_x$  for context and  $d_s$  for sense, while  $f$  and  $f_{\max}$  denote the original similarity function and its maximum, respectively.  $\alpha$  is a *standardization coefficient*, which is determined so that the mean squared distance be 1 in a dataset. According to this standardization, initial values of  $\sigma_x^2$ ,  $\sigma_s^2$  are always 1.

## 5 Evaluation

To confirm the effect of the proposed smoothing model and its combinatorial optimization scheme, we conducted WSD evaluations. The primary evaluations compare our method with conventional ones, in Section 5.2. Supplementary evaluations are described in the subsequent sections that include the comparison with SemEval-2 participating systems, and the analysis of model dynamics with the experimental data.

### 5.1 Evaluation Scheme

To make the evaluation comparable to state-of-the-art systems, we used the official dataset of the SemEval-2 English all-words WSD task (Agirre et al., 2010), which is currently the latest public dataset available with published results. The dataset consists of test data and background documents of the same *environment* domain. The test data consists of 1,398 target words (1,032 nouns and 366 verbs) in 5.3K running words. The background documents consists of 2.7M running words, which was used to compute distributional similarity.

Precisions and recalls were all computed using the official evaluation tool `scorer2` in fine-grained measure. The tool accepts answers either in probabilistic format (senses with probabilities for each target word) or in deterministic format (most likely senses, with no score information). As the proposed method is a probability model, we evaluated in the probabilistic way unless explicitly noted otherwise. For this reason, we evaluated all the sense probabilities as they were. Disambiguations were executed in separate runs for nouns and verbs, because no interaction takes place across POS in this metric implementation. The two runs'

results were combined later to a single answer to be input to `scorer2`.

The context metric space was composed by  $k$ -nearest neighbor words of distributional similarity (Lin, 1998), as is described in Section 4. The value of  $k$  was evaluated for  $\{2, 3, 5, 10, 20, 30, 50, 100, 200, 300\}$ . As for sense metric space, we evaluated two measures i.e., (Jiang and Conrath, 1997) denoted as *JCN*, and (Lesk, 1986) denoted as *Lesk*. In every condition, stopping criterion of iteration is always the number of iteration (500 times), irrespective of the convergence in likelihood.

Primary evaluations compared our method with two conventional methods. Those methods differ to ours only in scoring schemes. The first one is the method by McCarthy et al. (2004), which determines the word sense based on sense similarity and distributional similarity to the  $k$ -nearest neighbor words of a target word by distributional similarity. Our major advantage is the combinatorial optimization framework, while the conventional one employs word-by-word scheme. The second one is based on the method by Patwardhan et al. (2007), which determines the word sense by maximizing the sum of sense similarity to the  $k$  immediate neighbor words of a target word. The  $k$  words were forced to be selected from other target words of the same POS to the word of interest, so as to make information resource equivalent to the other comparable two methods. It is also a word-by-word method. It exploits no distributional similarity. Our major advantages are the combinatorial optimization scheme and the smoothing model to integrate distributional similarity. In the following section, these comparative methods are referred to as *Mc2004* and *Pat2007*, respectively.

### 5.2 Comparison with Conventional Methods

Let us first confirm our advantages compared to the conventional methods of *Mc2004* and *Pat2007*. The comparative results are shown in Figure 3 in recall measure. Precisions are simply omitted because the difference to the recalls are always the number of failures on referring to WordNet by mislabeling of lemmata or POSs, which is always the same for the three methods. Vertical range depicts 95% confidence intervals. The graphs also indicate the most-frequent-sense (MFS) baseline estimated from out-of-domain corpora, whose recall is 0.505 (Agirre et al., 2010).

As we can see in Figure 3(a) and 3(b), higher

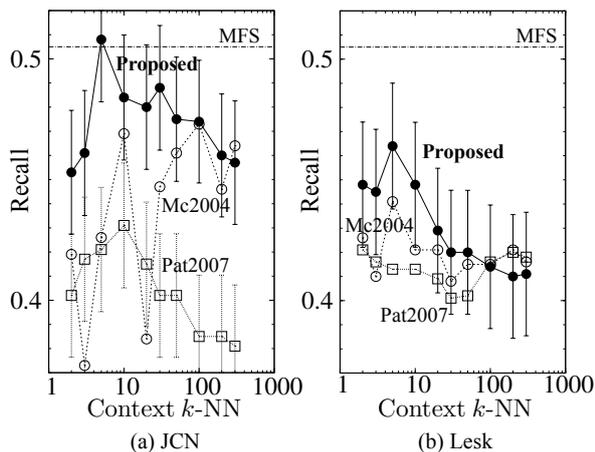


Figure 3: Comparison to the conventional methods that differ to our method only in scoring schemes.

Table 1: Comparison with the top-5 knowledge-based systems in SemEval-2 (JCN/ $k=5$ ).

Rank	Participants	R	P	Rn	Rv
-	<b>Proposed (best)</b>	<b>50.8</b>	<b>51.0</b>	<b>52.5</b>	<b>46.2</b>
-	<i>MFS Baseline</i>	50.5	50.5	52.7	44.3
1	Kulkarni et al. (2010)	49.5	51.2	51.6	43.4
2	Tran et al. (2010)	49.3	50.6	51.6	42.6
3	Tran et al. (2010)	49.1	50.4	51.5	42.5
4	Soroa et al. (2010)	48.1	48.1	48.7	46.2
5	Tran et al. (2010)	47.9	49.2	49.4	43.4
...	...	...	...	...	...
-	<i>Random Baseline</i>	23.2	23.2	25.3	17.2

recalls are obtained in the order of the proposed method, Mc2004, and Pat2007 on the whole. Comparing JCN and Lesk, difference among the three is smaller in Lesk. It is possibly because Lesk is a score not normalized for different word pairs, which makes the effect of distributional similarity unsteady especially when combining many  $k$ -nearest words. Therefore the recalls are expected to improve if proper normalization is applied to the proposed method and Mc2004. In JCN, the recalls of the proposed method significantly improve compared to Pat2007. Our best recall is 0.508 with JCN and  $k=5$ . Thus we can conclude that, though significance depends on metrics, our smoothing model and the optimization scheme are effective to improve accuracies.

### 5.3 Comparison with SemEval-2 Systems

We compared our best results with the participating systems of the task. Table 1 compares the details to the top-5 systems, which only includes unsupervised/knowledge-based ones and excludes supervised/weakly-supervised ones. Those values

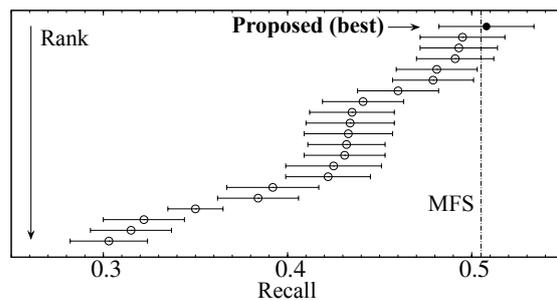


Figure 4: Comparison with the all 20 knowledge-based systems in SemEval-2 (JCN/ $k=5$ ).

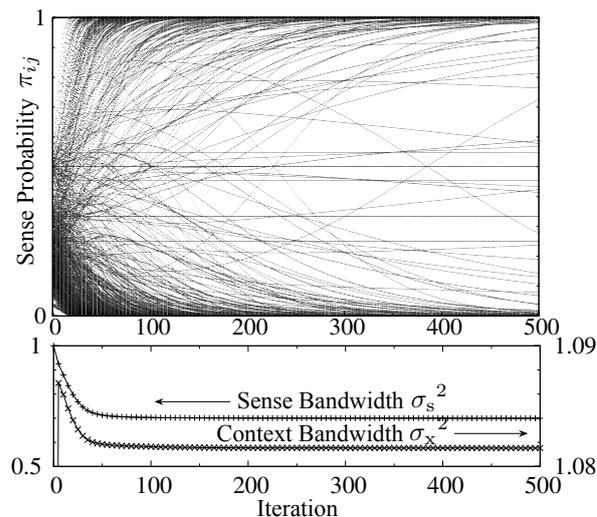


Figure 5: Model dynamics through iteration with SemEval-2 nouns (JCN/ $k=5$ ).

are transcribed from the official report (Agirre et al., 2010). “R” and “P” denote the recall and the precision for the whole dataset, while “Rn” and “Rv” denote the recall for nouns and verbs, respectively. The results are ranked by “R”, in accordance with the original report. As shown in the table, our best results outperform all of the systems and the MFS baseline.

Overall rankings are depicted in Figure 4. It maps our best results in the distribution of all the 20 unsupervised/knowledge-based participating systems. The ranges spreading left and right are 95% confidence intervals. As is seen from the figure, our best results are located above the top group, which are outside the confidence intervals of the other participants ranked intermediate or lower.

### 5.4 Analysis on Model Dynamics

This section examines the model dynamics with the SemEval-2 data, which has been illustrated

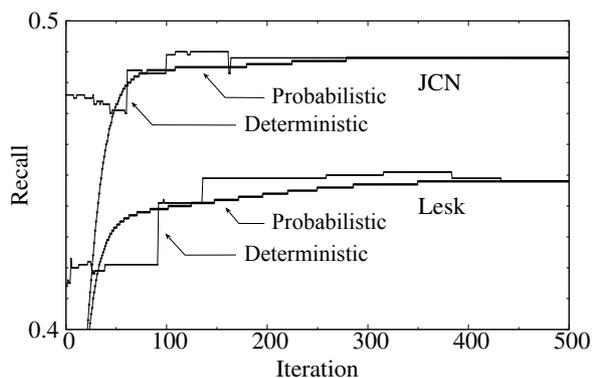


Figure 6: Recall improvement via iteration with SemEval-2 all POSs (JCN/ $k=30$ , Lesk/ $k=10$ ).

with pseudo data in Section 3.2. Let us start by looking at the upper half of Figure 5, which shows the change of sense probabilities through iteration. At the initial status (iteration 0), the probabilities were all  $1/2$  for bi-semous words, all  $1/3$  for tri-semous words, and so forth. As iteration proceeded, the probabilities gradually spread out to either side of 1 or 0, and finally at iteration 500, we can observe that almost all the words were clearly disambiguated. The lower half of Figure 5 shows the dynamics in bandwidths. Vertical axis on the left is for the sense bandwidth, and on the right is for the context bandwidth. We can observe those bandwidths became narrower as iteration proceeded. Intensity of smoothing was dynamically adjusted by the whole disambiguation status. These behaviors confirm that even with an actual dataset, it works as is expected, just as illustrated in Figure 2.

## 6 Discussion

This section discusses the validity of the proposed method as to i) sense-interdependent disambiguation and ii) reliability of data smoothing. We here analyze the second peak conditions at  $k = 30$  (JCN) and  $k = 10$  (Lesk) instead of the first peak at  $k = 5$ , because we can observe tendency the better with the larger number of word interactions.

### 6.1 Effects of Sense-interdependent Disambiguation

Let us first examine the effect of our sense-interdependent disambiguation. We would like to confirm that how the progressive disambiguation is carried out. Figure 6 shows the change of recall through iteration for JCN ( $k = 30$ ) and Lesk ( $k = 10$ ). Those recalls were obtained by

evaluating the status after each iteration. The recalls were here evaluated both in probabilistic format and in deterministic format. From the figure we can observe that the deterministic recalls also increased as well as the probabilistic recalls. This means that the ranks of sense candidates for each word were frequently altered through iteration, which further means that some new information not obtained earlier was delivered one after another to sense disambiguation for each word. From these results, we could confirm the expected sense-interdependency effect that a sense disambiguation of certain word affected to other words.

### 6.2 Reliability of Smoothing as Supervision

Let us now discuss the reliability of our smoothing model. In our method, sense disambiguation of a word is guided by its nearby words' extrapolation (smoothing). Sense accuracy fully depends on the reliability of the extrapolation. Generally speaking, statistical reliability increases as the number of random sampling increases. If we take sufficient number of *random words* as nearby words, the sense distribution comes close to the true distribution, and then we expect the statistically true sense distribution should find out the true sense of the target word, according to the *distributional hypotheses* (Harris, 1954). On the contrary, if we take nearby words that are biased to particular words, the sense distribution also becomes biased, and the extrapolation becomes less reliable.

We can compute the randomness of words that affect for sense disambiguation, by *word perplexity*. Let the word of interest be  $w \in V$ . The word perplexity is calculated as  $2^{H|w}$ , where  $H|w$  denotes the entropy defined as  $H|w \equiv -\sum_{w' \in V \setminus \{w\}} p(w'|w) \log_2 p(w'|w)$ . The conditional probability  $p(w'|w)$  denotes the probability with which a certain word  $w' \in V \setminus \{w\}$  determines the sense of  $w$ , which can be defined as the density ratio:  $p(w'|w) \propto \sum_{i: w_i=w} \sum_{i': w_{i'}=w'} \sum_{j, j'} Q_{i'j'} \mathcal{Q}_{ij}(h_{ij})$ .

The relation between word perplexity and probability change for ground-truth senses of nouns (JCN/ $k = 30$ ) is shown in Figure 7. The upper histogram shows the change in iteration 1-100, and the lower shows that of iteration 101-500. We divide the analysis at iteration 100, because roughly until the 100th iteration, the change in bandwidths converged, and the number of words to interact settled, as can be seen in Figure 5. The bars that

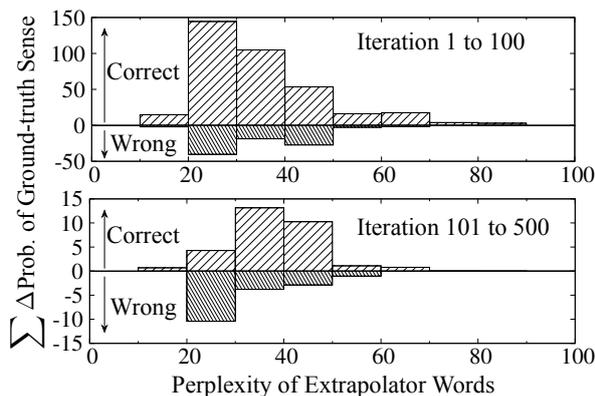


Figure 7: Correlation between reliability and perplexity with SemEval-2 nouns ( $JCN/k = 30$ ).

extend upward represent the sum of the amount raised (correct change), and the bars that extend downward represent the sum of the amount reduced (wrong change). From these figures, we observe that when perplexity is sufficiently large ( $\geq 30$ ), change occurred largely (79%) to the correct direction. In contrast, at the lower left of the figure, where perplexity is small ( $< 30$ ) and bandwidths has been narrowed at iteration 101-500, correct change occupied only 32% of the whole. Therefore, we can conclude that if sufficiently random samples of nearby words are provided, our smoothing model is reliable, though it is trained in an unsupervised fashion.

## 7 Related Work

As described in Section 1, graph-based WSD has been extensively studied, since graphs are favorable structure to deal with interactions of data on vertices. Conventional studies typically consider as vertices the instances of input or target class, e.g. knowledge-based approaches typically regard *senses as vertices* (see Section 1), and corpus-based approaches such as (Véronis, 2004) regard *words as vertices* or (Niu et al., 2005) regards *context as vertices*. Our method can be viewed as one of graph-based methods, but it regards *input-to-class mapping as vertices*, and the edges represent the relations both together in context and in sense. Mihalcea (2005) proposed graph-based methods, whose vertices are sense label hypotheses on word sequence. Our method generalize context representation.

In the evaluation, our method was compared to SemEval-2 systems. The main subject of the SemEval-2 task was domain adaptation, therefore

those systems each exploited their own adaptation techniques. Kulkarni et al. (2010) used a WordNet pre-pruning. Disambiguation is performed by considering only those candidate synsets that belong to the top- $k$  largest connected components of the WordNet on domain corpus. Tran et al. (2010) used over 3TB domain documents acquired by Web search. They parsed those documents and extracted the statistics on dependency relation for disambiguation. Soroa et al. (2010) used the method by Agirre et al. (2009) described in Section 1. They disambiguated each target word using its distributionally similar words instead of its immediate context words.

The proposed method is an extension of density estimation (Parzen, 1962), which is a construction of an estimate based on *observed data*. Our method naturally extends the density estimation in two points, which make it applicable to unsupervised knowledge-based WSD. First, we introduce stochastic treatment of data, which is no longer observations but hypotheses having ambiguity. This extension makes the hypotheses possible to cross-validate the plausibility each other. Second, we extend the definition of density from Euclidean distance to general metric, which makes the proposed method applicable to a wide variety of corpus-based context similarities and dictionary-based sense similarities.

## 8 Conclusions

We proposed a novel *smoothing model* with a *combinatorial optimization scheme* for all-words WSD from untagged corpora. Experimental results showed that our method significantly improves the accuracy of conventional methods by exceeding most-frequent-sense baseline performance where none of SemEval-2 unsupervised systems reached. Detailed inspection of dynamics clearly show that the proposed optimization method effectively exploit the sense-dependency of all-words. Moreover, our smoothing model, though unsupervised, provides reliable supervision when sufficiently random samples of words are available as nearby words. Thus it was confirmed that this method is valid for finding the optimal combination of word senses with large untagged corpora. We hope this study would elicit further investigation in this important area.

## References

- Eneko Agirre and Philip Edmonds. 2006. *Word sense disambiguation: Algorithms and applications*, volume 33. Springer Science+ Business Media.
- Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41.
- Eneko Agirre, Oier Lopez De Lacalle, Aitor Soroa, and Informatika Fakultatea. 2009. Knowledge-based wsd on specific domains: performing better than generic supervised wsd. In *Proceedings of the 21st international joint conference on Artificial intelligence*, pages 1501–1506.
- Eneko Agirre, Oier Lopez de Lacalle, Christiane Fellbaum, Shu-Kai Hsieh, Maurizio Tesconi, Monica Monachini, Piek Vossen, and Roxanne Segers. 2010. Semeval-2010 task 17: All-words word sense disambiguation on a specific domain. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 75–80.
- Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the rasp system. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 77–80.
- Arthur Pentland Dempster, Nan McKenzie Laird, and Donald Bruce Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.
- Zellig Sabbetai Harris. 1954. Distributional structure. *Word*.
- Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*.
- Anup Kulkarni, Mitesh M. Khapra, Saurabh Sohoney, and Pushpak Bhattacharyya. 2010. CFILT: Resource conscious approaches for all-words domain specific. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 421–426.
- Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774.
- Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 279–286.
- Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2007. Unsupervised acquisition of predominant word senses. *Computational Linguistics*, 33(4):553–590.
- Rada Mihalcea. 2005. Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 411–418.
- Roberto Navigli and Mirella Lapata. 2007. Graph connectivity measures for unsupervised word sense disambiguation. In *Proceedings of the 20th international joint conference on Artificial intelligence*, pages 1683–1688.
- Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.
- Zheng-Yu Niu, Dong-Hong Ji, and Chew Lim Tan. 2005. Word sense disambiguation using label propagation based semi-supervised learning. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 395–402.
- Emanuel Parzen. 1962. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.
- Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. 2007. UMND1: Unsupervised word sense disambiguation using contextual semantic relatedness. In *proceedings of the 4th International Workshop on Semantic Evaluations*, pages 390–393.
- Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. WordNet::Similarity: measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, pages 38–41.
- Aitor Soroa, Eneko Agirre, Oier Lopez de Lacalle, Monica Monachini, Jessie Lo, Shu-Kai Hsieh, Wauter Bosma, and Piek Vossen. 2010. Kyoto: An integrated system for specific domain WSD. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 417–420.
- Andrew Tran, Chris Bowes, David Brown, Ping Chen, Max Choly, and Wei Ding. 2010. TreeMatch: A fully unsupervised WSD system using dependency knowledge on a specific domain. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 396–401.
- Jean Véronis. 2004. HyperLex: lexical cartography for information retrieval. *Computer Speech & Language*, 18(3):223–252.

# The Role of Syntax in Vector Space Models of Compositional Semantics

Karl Moritz Hermann and Phil Blunsom

Department of Computer Science

University of Oxford

Oxford, OX1 3QD, UK

{karl.moritz.hermann, phil.blunsom}@cs.ox.ac.uk

## Abstract

Modelling the compositional process by which the meaning of an utterance arises from the meaning of its parts is a fundamental task of Natural Language Processing. In this paper we draw upon recent advances in the learning of vector space representations of sentential semantics and the transparent interface between syntax and semantics provided by Combinatory Categorical Grammar to introduce Combinatory Categorical Autoencoders. This model leverages the CCG combinatory operators to guide a non-linear transformation of meaning within a sentence. We use this model to learn high dimensional embeddings for sentences and evaluate them in a range of tasks, demonstrating that the incorporation of syntax allows a concise model to learn representations that are both effective and general.

## 1 Introduction

Since Frege stated his ‘Principle of Semantic Compositionality’ in 1892 researchers have pondered both how the meaning of a complex expression is determined by the meanings of its parts, and how those parts are combined. (Frege, 1892; Pelletier, 1994). Over a hundred years on the choice of representational unit for this process of compositional semantics, and how these units combine, remain open questions.

Frege’s principle may be debatable from a linguistic and philosophical standpoint, but it has provided a basis for a range of formal approaches to semantics which attempt to capture meaning in logical models. The Montague grammar (Montague, 1970) is a prime example for this, building a model of composition based on lambda-calculus and formal logic. More recent work

in this field includes the Combinatory Categorical Grammar (CCG), which also places increased emphasis on syntactic coverage (Szabolcsi, 1989).

Recently those searching for the right representation for compositional semantics have drawn inspiration from the success of distributional models of lexical semantics. This approach represents single words as distributional vectors, implying that a word’s meaning is a function of the environment it appears in, be that its syntactic role or co-occurrences with other words (Pereira et al., 1993; Schütze, 1998). While distributional semantics is easily applied to single words, sparsity implies that attempts to directly extract distributional representations for larger expressions are doomed to fail. Only in the past few years has it been attempted to extend these representations to semantic composition. Most approaches here use the idea of vector-matrix composition to learn larger representations from single-word encodings (Baroni and Zamparelli, 2010; Grefenstette and Sadrzadeh, 2011; Socher et al., 2012b). While these models have proved very promising for compositional semantics, they make minimal use of linguistic information beyond the word level.

In this paper we bridge the gap between recent advances in machine learning and more traditional approaches within computational linguistics. We achieve this goal by employing the CCG formalism to consider compositional structures at any point in a parse tree. CCG is attractive both for its transparent interface between syntax and semantics, and a small but powerful set of combinatory operators with which we can parametrise our non-linear transformations of compositional meaning.

We present a novel class of recursive models, the Combinatory Categorical Autoencoders (CCA), which marry a semantic process provided by a recursive autoencoder with the syntactic representations of the CCG formalism. Through this model we seek to answer two ques-



representations have then successfully been applied to a number of tasks including word sense disambiguation (Schütze, 1998) and selectional preference (Pereira et al., 1993; Lin, 1999).

While it is theoretically possible to apply the same mechanism to larger expressions, sparsity prevents learning meaningful distributional representations for expressions much larger than single words.<sup>2</sup>

Vector space models of compositional semantics aim to fill this gap by providing a methodology for deriving the representation of an expression from those of its parts. While distributional representations frequently serve to encode single words in such approaches this is not a strict requirement.

There are a number of ideas on how to define composition in such vector spaces. A general framework for semantic vector composition was proposed in Mitchell and Lapata (2008), with Mitchell and Lapata (2010) and more recently Blacoe and Lapata (2012) providing good overviews of this topic. Notable approaches to this issue include Baroni and Zamparelli (2010), who compose nouns and adjectives by representing them as vectors and matrices, respectively, with the compositional representation achieved by multiplication. Grefenstette and Sadrzadeh (2011) use a similar approach with matrices for relational words and vectors for arguments. These two approaches are combined in Grefenstette et al. (2013), producing a tensor-based semantic framework with tensor contraction as composition operation.

Another set of models that have very successfully been applied in this area are recursive autoencoders (Socher et al., 2011a; Socher et al., 2011b), which are discussed in the next section.

### 2.3 Recursive Autoencoders

Autoencoders are a useful tool to compress information. One can think of an autoencoder as a funnel through which information has to pass (see Figure 2). By forcing the autoencoder to reconstruct an input given only the reduced amount of information available inside the funnel it serves as a compression tool, representing high-dimensional objects in a lower-dimensional space.

Typically a given autoencoder, that is the functions for encoding and reconstructing data, are

<sup>2</sup>The experimental setup in (Baroni and Zamparelli, 2010) is one of the few examples where distributional representations are used for word pairs.

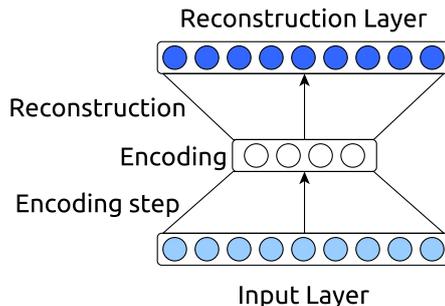


Figure 2: A simple three-layer autoencoder. The input represented by the vector at the bottom is being encoded in a smaller vector (middle), from which it is then reconstructed (top) into the same dimensionality as the original input vector.

used on multiple inputs. By optimizing the two functions to minimize the difference between all inputs and their respective reconstructions, this autoencoder will effectively discover some hidden structures within the data that can be exploited to represent it more efficiently.

As a simple example, assume input vectors  $x_i \in \mathbb{R}^n, i \in (0..N)$ , weight matrices  $W^{enc} \in \mathbb{R}^{(m \times n)}$ ,  $W^{rec} \in \mathbb{R}^{(n \times m)}$  and biases  $b^{enc} \in \mathbb{R}^m$ ,  $b^{rec} \in \mathbb{R}^n$ . The encoding matrix and bias are used to create an encoding  $e_i$  from  $x_i$ :

$$e_i = f^{enc}(x_i) = W^{enc}x_i + b^{enc} \quad (1)$$

Subsequently  $e \in \mathbb{R}^m$  is used to reconstruct  $x$  as  $x'$  using the reconstruction matrix and bias:

$$x'_i = f^{rec}(e_i) = W^{rec}e_i + b^{rec} \quad (2)$$

$\theta = (W^{enc}, W^{rec}, b^{enc}, b^{rec})$  can then be learned by minimizing the error function describing the difference between  $x'$  and  $x$ :

$$E = \frac{1}{2} \sum_i^N \|x'_i - x_i\|^2 \quad (3)$$

Now, if  $m < n$ , this will intuitively lead to  $e_i$  encoding a latent structure contained in  $x_i$  and shared across all  $x_j, j \in (0..N)$ , with  $\theta$  encoding and decoding to and from that hidden structure.

It is possible to apply multiple autoencoders on top of each other, creating a deep autoencoder (Bengio et al., 2007; Hinton and Salakhutdinov, 2006). For such a multi-layered model to learn anything beyond what a single layer could learn, a non-linear transformation  $g$  needs to be applied at each layer. Usually, a variant of the sigmoid ( $\sigma$ )

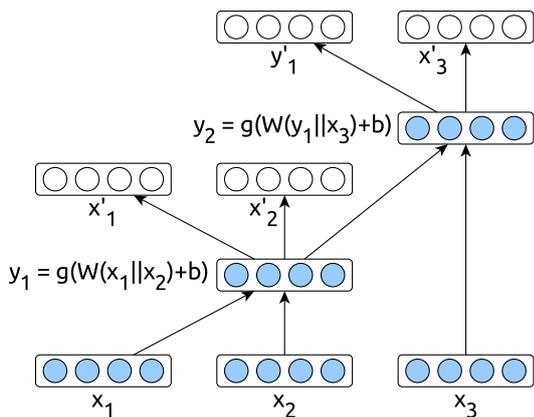


Figure 3: RAE with three inputs. Vectors with filled (blue) circles represent input and hidden units; blanks (white) denote reconstruction layers.

or hyperbolic tangent ( $\tanh$ ) function is used for  $g$  (LeCun et al., 1998).

$$\begin{aligned} f^{enc}(x_i) &= g(W^{enc}x_i + b^{enc}) \\ f^{rec}(e_i) &= g(W^{rec}e_i + b^{rec}) \end{aligned} \quad (4)$$

Furthermore, autoencoders can easily be used as a composition function by concatenating two input vectors, such that:

$$\begin{aligned} e &= f(x_1, x_2) = g(W(x_1||x_2) + b) \\ (x'_1||x'_2) &= g(W'e + b') \end{aligned} \quad (5)$$

Extending this idea, recursive autoencoders (RAE) allow the modelling of data of variable size. By setting the  $n = 2m$ , it is possible to recursively combine a structure into an autoencoder tree. See Figure 3 for an example, where  $x_1, x_2, x_3$  are recursively encoded into  $y_2$ .

The recursive application of autoencoders was first introduced in Pollack (1990), whose recursive auto-associative memories learn vector representations over pre-specified recursive data structures. More recently this idea was extended and applied to dynamic structures (Socher et al., 2011b).

These types of models have become increasingly prominent since developments within the field of Deep Learning have made the training of such hierarchical structures more effective and tractable (LeCun et al., 1998; Hinton et al., 2006).

Intuitively the top layer of an RAE will encode aspects of the information stored in all of the input vectors. Previously, RAE have successfully been applied to a number of tasks including sentiment analysis, paraphrase detection, relation extraction

Model	CCG Elements
CCAIE-A	parse
CCAIE-B	parse + rules
CCAIE-C	parse + rules + types
CCAIE-D	parse + rules + child types

Table 1: Aspects of the CCG formalism used by the different models explored in this paper.

and 3D object identification (Blacoe and Lapata, 2012; Socher et al., 2011b; Socher et al., 2012a).

### 3 Model

The models in this paper combine the power of recursive, vector-based models with the linguistic intuition of the CCG formalism. Their purpose is to learn semantically meaningful vector representations for sentences and phrases of variable size, while the purpose of this paper is to investigate the use of syntax and linguistic formalisms in such vector-based compositional models.

We assume a CCG parse to be given. Let  $C$  denote the set of combinatory rules, and  $T$  the set of categories used, respectively. We use the parse tree to structure an RAE, so that each combinatory step is represented by an autoencoder function. We refer to these models Categorical Combinatory Autoencoders (CCAIE). In total this paper describes four models making increasing use of the CCG formalism (see table 1).

As an internal baseline we use model CCAIE-A, which is an RAE structured along a CCG parse tree. CCAIE-A uses a single weight matrix each for the encoding and reconstruction step (see Table 2). This model is similar to Socher et al. (2011b), except that we use a fixed structure in place of the greedy tree building approach. As CCAIE-A uses only minimal syntactic guidance, this should allow us to better ascertain to what degree the use of syntax helps our semantic models.

Our second model (CCAIE-B) uses the composition function in equation (6), with  $c \in C$ .

$$\begin{aligned} f^{enc}(x, y, c) &= g(W_{enc}^c(x||y) + b_{enc}^c) \\ f^{rec}(e, c) &= g(W_{rec}^c e + b_{rec}^c) \end{aligned} \quad (6)$$

This means that for every combinatory rule we define an equivalent autoencoder composition function by parametrizing both the weight matrix and bias on the combinatory rule (e.g. Figure 4).

In this model, as in the following ones, we assume a reconstruction step symmetric with the

Model	Encoding Function
CCAIE-A	$f(x, y) = g(W(x  y) + b)$
CCAIE-B	$f(x, y, c) = g(W^c(x  y) + b^c)$
CCAIE-C	$f(x, y, c, t) = g\left(\sum_{p \in \{c, t\}} (W^p(x  y) + b^p)\right)$
CCAIE-D	$f(x, y, c, tx, ty) = g(W^c(W^{tx}x + W^{ty}y) + b^c)$

Table 2: Encoding functions of the four CCAIE models discussed in this paper.

$$\frac{\alpha : X/Y \quad \beta : Y}{\alpha\beta : X} \rightarrow g(W_{enc}^>(\alpha||\beta) + b_{enc}^>)$$

Figure 4: Forward application as CCG combinator and autoencoder rule respectively.

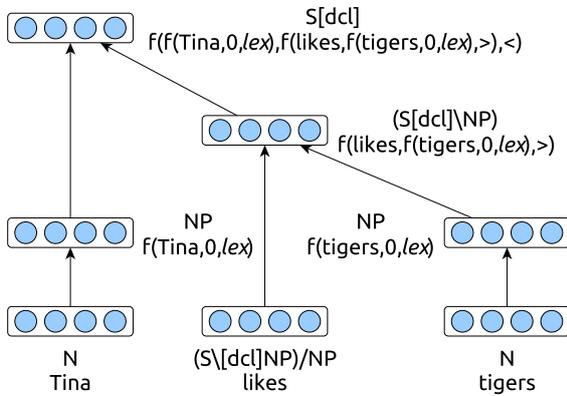


Figure 5: CCAIE-B applied to *Tina likes tigers*. Next to each vector are the CCG category (top) and the word or function representing it (bottom). *lex* describes the unary type-changing operation.  $>$  and  $<$  are forward and backward application.

composition step. For the remainder of this paper we will focus on the composition step and drop the use of *enc* and *rec* in variable names where it isn't explicitly required. Figure 5 shows model CCAIE-B applied to our previous example sentence.

While CCAIE-B uses only the combinatory rules, we want to make fuller use of the linguistic information available in CCG. For this purpose, we build another model CCAIE-C, which parametrizes on both the combinatory rule  $c \in C$  and the CCG category  $t \in T$  at every step (see Figure 2). This model provides an additional degree of insight, as the categories  $T$  are semantically and syntactically more expressive than the CCG combinatory rules by themselves. Summing over weights parametrised on  $c$  and  $t$  respectively, adds an additional degree of freedom and also al-

lows for some model smoothing.

An alternative approach is encoded in model CCAIE-D. Here we consider the categories not of the element represented, but of the elements it is generated from together with the combinatory rule applied to them. The intuition is that in the first step we transform two expressions based on their syntax. Subsequently we combine these two conditioned on their joint combinatory rule.

## 4 Learning

In this section we briefly discuss unsupervised learning for our models. Subsequently we describe how these models can be extended to allow for semi-supervised training and evaluation.

Let  $\theta = (W, B, L)$  be our model parameters and  $\lambda$  a vector with regularization parameters for all model parameters.  $W$  represents the set of all weight matrices,  $B$  the set of all biases and  $L$  the set of all word vectors. Let  $N$  be the set of training data consisting of tree-nodes  $n$  with inputs  $x_n, y_n$  and reconstruction  $r_n$ . The error given  $n$  is:

$$E(n|\theta) = \frac{1}{2} \left\| r_n - (x_n || y_n) \right\|^2 \quad (7)$$

The gradient of the regularised objective function then becomes:

$$\frac{\partial J}{\partial \theta} = \frac{1}{N} \sum_n \frac{\partial E(n|\theta)}{\partial \theta} + \lambda \theta \quad (8)$$

We learn the gradient using backpropagation through structure (Goller and Küchler, 1996), and minimize the objective function using L-BFGS.

For more details about the partial derivatives used for backpropagation, see the documentation accompanying our model implementation.<sup>3</sup>

<sup>3</sup><http://www.karlmoritz.com/>

## 4.1 Supervised Learning

The unsupervised method described so far learns a vector representation for each sentence. Such a representation can be useful for some tasks such as paraphrase detection, but is not sufficient for other tasks such as sentiment classification, which we are considering in this paper.

In order to extract sentiment from our models, we extend them by adding a supervised classifier on top, using the learned representations  $v$  as input for a binary classification model:

$$\text{pred}(l=1|v, \theta) = \text{sigmoid}(W_{\text{label}} v + b_{\text{label}}) \quad (9)$$

Given our corpus of CCG parses with label pairs  $(N, l)$ , the new objective function becomes:

$$J = \frac{1}{N} \sum_{(N,l)} E(N, l, \theta) + \frac{\lambda}{2} \|\theta\|^2 \quad (10)$$

Assuming each node  $n \in N$  contains children  $x_n, y_n$ , encoding  $e_n$  and reconstruction  $r_n$ , so that  $n = \{x, y, e, r\}$  this breaks down into:

$$E(N, l, \theta) = \sum_{n \in N} \alpha E_{\text{rec}}(n, \theta) + (1 - \alpha) E_{\text{lbl}}(e_n, l, \theta) \quad (11)$$

$$E_{\text{rec}}(n, \theta) = \frac{1}{2} \left\| [x_n \| y_n] - r_n \right\|^2 \quad (12)$$

$$E_{\text{lbl}}(e, l, \theta) = \frac{1}{2} \|l - e\|^2 \quad (13)$$

This method of introducing a supervised aspect to the autoencoder largely follows the model described in Socher et al. (2011b).

## 5 Experiments

We describe a number of standard evaluations to determine the comparative performance of our model. The first task of sentiment analysis allows us to compare our CCG-conditioned RAE with similar, existing models. In a second experiment, we apply our model to a compound similarity evaluation, which allows us to evaluate our models against a larger class of vector-based models (Blacoe and Lapata, 2012). We conclude with some qualitative analysis to get a better idea of whether the combination of CCG and RAE can learn semantically expressive embeddings.

In our experiments we use the hyperbolic tangent as nonlinearity  $g$ . Unless stated otherwise we

use word-vectors of size 50, initialized using the embeddings provided by Turian et al. (2010) based on the model of Collobert and Weston (2008).<sup>4</sup>

We use the C&C parser (Clark and Curran, 2007) to generate CCG parse trees for the data used in our experiments. For models CCAE-C and CCAE-D we use the 25 most frequent CCG categories (as extracted from the British National Corpus) with an additional general weight matrix in order to catch all remaining types.

### 5.1 Sentiment Analysis

We evaluate our model on the MPQA opinion corpus (Wiebe et al., 2005), which annotates expressions for sentiment.<sup>5</sup> The corpus consists of 10,624 instances with approximately 70 percent describing a negative sentiment. We apply the same pre-processing as (Nakagawa et al., 2010) and (Socher et al., 2011b) by using an additional sentiment lexicon (Wilson et al., 2005) during the model training for this experiment.

As a second corpus we make use of the sentence polarity (SP) dataset v1.0 (Pang and Lee, 2005).<sup>6</sup> This dataset consists of 10662 sentences extracted from movie reviews which are manually labelled with positive or negative sentiment and equally distributed across sentiment.

**Experiment 1: Semi-Supervised Training** In the first experiment, we use the semi-supervised training strategy described previously and initialize our models with the embeddings provided by Turian et al. (2010). The results of this evaluation are in Table 3. While we achieve the best performance on the MPQA corpus, the results on the SP corpus are less convincing. Perhaps surprisingly, the simplest model CCAE-A outperforms the other models on this dataset.

When considering the two datasets, sparsity seems a likely explanation for this difference in results: In the MPQA experiment most instances are very short with an average length of 3 words, while the average sentence length in the SP corpus is 21 words. The MPQA task is further simplified through the use of an additional sentiment lexicon. Considering dictionary size, the SP corpus has a dictionary of 22k words, more than three times the size of the MPQA dictionary.

<sup>4</sup><http://www.metaoptimize.com/projects/wordreprs/>

<sup>5</sup><http://mpqa.cs.pitt.edu/>

<sup>6</sup><http://www.cs.cornell.edu/people/pabo/movie-review-data/>

Method	MPQA	SP
Voting with two lexica	81.7	63.1
MV-RNN (Socher et al., 2012b)	-	79.0
RAE (rand) (Socher et al., 2011b)	85.7	76.8
TCRF (Nakagawa et al., 2010)	86.1	77.3
RAE (init) (Socher et al., 2011b)	86.4	77.7
NB (Wang and Manning, 2012)	86.7	79.4
CCAЕ-A	86.3	77.8
CCAЕ-B	87.1	77.1
CCAЕ-C	87.1	77.3
CCAЕ-D	<b>87.2</b>	76.7

Table 3: Accuracy of sentiment classification on the sentiment polarity (SP) and MPQA datasets. For NB we only display the best result among a larger group of models analysed in that paper.

This issue of sparsity is exacerbated in the more complex CCAE models, where the training points are spread across different CCG types and rules. While the initialization of the word vectors with previously learned embeddings (as was previously shown by Socher et al. (2011b)) helps the models, all other model variables such as composition weights and biases are still initialised randomly and thus highly dependent on the amount of training data available.

**Experiment 2: Pretraining** Due to our analysis of the results of the initial experiment, we ran a second series of experiments on the SP corpus. We follow (Scheible and Schütze, 2013) for this second series of experiments, which are carried out on a random 90/10 training-testing split, with some data reserved for development.

Instead of initialising the model with external word embeddings, we first train it on a large amount of data with the aim of overcoming the sparsity issues encountered in the previous experiment. Learning is thus divided into two steps:

The first, unsupervised training phase, uses the British National Corpus together with the SP corpus. In this phase only the reconstruction signal is used to learn word embeddings and transformation matrices. Subsequently, in the second phase, only the SP corpus is used, this time with both the reconstruction and the label error.

By learning word embeddings and composition matrices on more data, the model is likely to generalise better. Particularly for the more complex models, where the composition functions are conditioned on various CCG parameters, this should

Model	Training	
	Regular	Pretraining
CCAЕ-A	77.8	79.5
CCAЕ-B	76.9	79.8
CCAЕ-C	77.1	81.0
CCAЕ-D	76.9	79.7

Table 4: Effect of pretraining on model performance on the SP dataset. Results are reported on a random subsection of the SP corpus; thus numbers for the regular training method differ slightly from those in Table 3.

help to overcome issues of sparsity.

If we consider the results of the pre-trained experiments in Table 4, this seems to be the case. In fact, the trend of the previous results has been reversed, with the more complex models now performing best, whereas in the previous experiments the simpler models performed better. Using the Turian embeddings instead of random initialisation did not improve results in this setup.

## 5.2 Compound Similarity

In a second experiment we use the dataset from Mitchell and Lapata (2010) which contains similarity judgements for adjective-noun, noun-noun and verb-object pairs.<sup>7</sup> All compound pairs have been ranked for semantic similarity by a number of human annotators. The task is thus to rank these pairs of word pairs by their semantic similarity.

For instance, the two compounds *vast amount* and *large quantity* are given a high similarity score by the human judges, while *northern region* and *early age* are assigned no similarity at all.

We train our models as fully unsupervised autoencoders on the British National Corpus for this task. We assume fixed parse trees for all of the compounds (Figure 6), and use these to compute compound level vectors for all word pairs. We subsequently use the cosine distance between each compound pair as our similarity measure. We use Spearman’s rank correlation coefficient ( $\rho$ ) for evaluation; hence there is no need to rescale our scores (-1.0 – 1.0) to the original scale (1.0 – 7.0).

Blacoe and Lapata (2012) have an extensive comparison of the performance of various vector-based models on this data set to which we compare our model in Table 5. The CCAE models outper-

<sup>7</sup><http://homepages.inf.ed.ac.uk/mlap/resources/index.html>

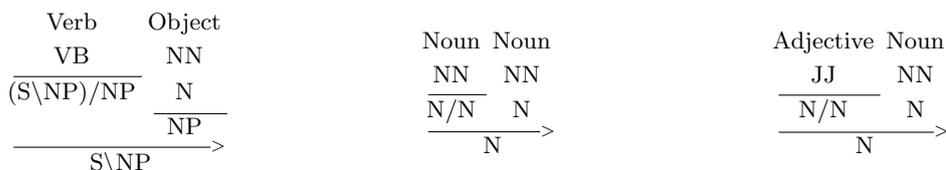


Figure 6: Assumed CCG parse structure for the compound similarity evaluation.

Method	Adj-N	N-N	V-Obj
Human	0.52	0.49	0.55
(Blacoe and Lapata, 2012)			
⊙/+	0.21 - 0.48	0.22 - 0.50	0.18 - 0.35
RAE	0.19 - 0.31	0.24 - 0.30	0.09 - 0.28
CCAЕ-B	0.38	0.44	0.34
CCAЕ-C	0.38	0.41	0.23
CCAЕ-D	0.41	0.44	0.29

Table 5: Correlation coefficients of model predictions for the compound similarity task. Numbers show Spearman’s rank correlation coefficient ( $\rho$ ). Higher numbers indicate better correlation.

form the RAE models provided by Blacoe and Lapata (2012), and score towards the upper end of the range of other models considered in that paper.

### 5.3 Qualitative Analysis

To get better insight into our models we also perform a small qualitative analysis. Using one of the models trained on the MPQA corpus, we generate word-level representations of all phrases in this corpus and subsequently identify the most related expressions by using the cosine distance measure. We perform this experiment on all expressions of length 5, considering all expressions with a word length between 3 and 7 as potential matches.

As can be seen in Table 6, this works with varying success. Linking expressions such as *conveying the message of peace* and *safeguard(ing) peace and security* suggests that the model does learn some form of semantics.

On the other hand, the connection between *expressed their satisfaction and support* and *expressed their admiration and surprise* suggests that the pure word level content still has an impact on the model analysis. Likewise, the expressions *is a story of success* and *is a staunch supporter* have some lexical but little semantic overlap. Further reducing this link between the lexical and the semantic representation is an issue that should be addressed in future work in this area.

## 6 Discussion

Overall, our models compare favourably with the state of the art. On the MPQA corpus model CCAE-D achieves the best published results we are aware of, whereas on the SP corpus we achieve competitive results. With an additional, unsupervised training step we achieved results beyond the current state of the art on this task, too.

**Semantics** The qualitative analysis and the experiment on compounds demonstrate that the CCAE models are capable of learning semantics. An advantage of our approach—and of autoencoders generally—is their ability to learn in an unsupervised setting. The pre-training step for the sentiment task was essentially the same training step as used in the compound similarity task. While other models such as the MV-RNN (Socher et al., 2012b) achieve good results on a particular task, they do not allow unsupervised training. This prevents the possibility of pretraining, which we showed to have a big impact on results, and further prevents the training of general models: The CCAE models can be used for multiple tasks without the need to re-train the main model.

**Complexity** Previously in this paper we argued that our models combined the strengths of other approaches. By using a grammar formalism we increase the expressive power of the model while the complexity remains low. For the complexity analysis see Table 7. We strike a balance between the greedy approaches (e.g. Socher et al. (2011b)), where learning is quadratic in the length of each sentence and existing syntax-driven approaches such as the MV-RNN of Socher et al. (2012b), where the size of the model, that is the number of variables that needs to be learned, is quadratic in the size of the word-embeddings.

**Sparsity** Parametrizing on CCG types and rules increases the size of the model compared to a greedy RAE (Socher et al., 2011b). The effect of this was highlighted by the sentiment analysis task, with the more complex models performing

Expression	Most Similar
convey the message of peace	safeguard peace and security
keep alight the flame of	keep up the hope
has a reason to repent	has no right
a significant and successful strike	a much better position
it is reassuring to believe	it is a positive development
expressed their satisfaction and support	expressed their admiration and surprise
is a story of success	is a staunch supporter
are lining up to condemn	are going to voice their concerns
more sanctions should be imposed	charges being leveled
could fray the bilateral goodwill	could cause serious damage

Table 6: Phrases from the MPQA corpus and their semantically closest match according to CCAE-D.

Model	Complexity	
	Size	Learning
MV-RNN	$\mathcal{O}(nw^2)$	$\mathcal{O}(l)$
RAE	$\mathcal{O}(nw)$	$\mathcal{O}(l^2)$
CCAЕ-*	$\mathcal{O}(nw)$	$\mathcal{O}(l)$

Table 7: Comparison of models.  $n$  is dictionary size,  $w$  embedding width,  $l$  is sentence length. We can assume  $l \gg n \gg w$ . Additional factors such as CCG rules and types are treated as small constants for the purposes of this analysis.

worse in comparison with the simpler ones. We were able to overcome this issue by using additional training data. Beyond this, it would also be interesting to investigate the relationships between different types and to derive functions to incorporate this into the learning procedure. For instance model learning could be adjusted to enforce some mirroring effects between the weight matrices of forward and backward application, or to support similarities between those of forward application and composition.

**CCG-Vector Interface** Exactly how the information contained in a CCG derivation is best applied to a vector space model of compositionality is another issue for future research. Our investigation of this matter by exploring different model setups has proved somewhat inconclusive. While CCAE-D incorporated the deepest conditioning on the CCG structure, it did not decisively outperform the simpler CCAE-B which just conditioned on the combinatory operators. Issues of sparsity, as shown in our experiments on pretraining, have a significant influence, which requires further study.

## 7 Conclusion

In this paper we have brought a more formal notion of semantic compositionality to vector space models based on recursive autoencoders. This was achieved through the use of the CCG formalism to provide a conditioning structure for the matrix vector products that define the RAE.

We have explored a number of models, each of which conditions the compositional operations on different aspects of the CCG derivation. Our experimental findings indicate a clear advantage for a deeper integration of syntax over models that use only the bracketing structure of the parse tree.

The most effective way to condition the compositional operators on the syntax remains unclear. Once the issue of sparsity had been addressed, the complex models outperformed the simpler ones. Among the complex models, however, we could not establish significant or consistent differences to convincingly argue for a particular approach.

While the connections between formal linguistics and vector space approaches to NLP may not be immediately obvious, we believe that there is a case for the continued investigation of ways to best combine these two schools of thought. This paper represents one step towards the reconciliation of traditional formal approaches to compositional semantics with modern machine learning.

## Acknowledgements

We thank the anonymous reviewers for their feedback and Richard Socher for providing additional insight into his models. Karl Moritz would further like to thank Sebastian Riedel for hosting him at UCL while this paper was written. This work has been supported by the EPSRC.

## References

- Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of EMNLP*, pages 1183–1193.
- Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *Proceedings of LREC*, pages 3196–3200, Istanbul, Turkey.
- Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2007. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19*, pages 153–160.
- William Blacoe and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of EMNLP-CoNLL*, pages 546–556.
- Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with ccg and log-linear models. *CL*, 33(4):493–552, December.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*.
- James Curran, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-scale nlp with c&c and boxer. In *Proceedings of ACL Demo and Poster Sessions*, pages 33–36.
- J. R. Firth. 1957. A synopsis of linguistic theory 1930–55. 1952–59:1–32.
- Gottfried Frege. 1892. Über Sinn und Bedeutung. In Mark Textor, editor, *Funktion - Begriff - Bedeutung*, volume 4 of *Sammlung Philosophie*. Vandenhoeck & Ruprecht, Göttingen.
- Christoph Goller and Andreas Küchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of the ICNN-96*, pages 347–352. IEEE.
- Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of EMNLP*, pages 1394–1404.
- Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics.
- G. E. Hinton and R. R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.
- Geoffrey E. Hinton, Simon Osindero, Max Welling, and Yee Whye Teh. 2006. Unsupervised discovery of nonlinear structure using contrastive backpropagation. *Cognitive Science*, 30(4):725–731.
- Julia Hockenmaier and Mark Steedman. 2007. CCG-bank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *CL*, 33(3):355–396, September.
- Ray Jackendoff. 1972. *Semantic Interpretation in Generative Grammar*. MIT Press, Cambridge, MA.
- Marcus Kracht. 2008. Compositionality in Montague Grammar. In Edouard Machery und Markus Werning Wolfram Hinzen, editor, *Handbook of Compositionality*, pages 47 – 63. Oxford University Press.
- Yann LeCun, Leon Bottou, Genevieve Orr, and Klaus-Robert Müller. 1998. Efficient backprop. In G. Orr and Müller K., editors, *Neural Networks: Tricks of the trade*. Springer.
- Dekang Lin. 1999. Automatic identification of non-compositional phrases. In *Proceedings of ACL*, pages 317–324.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *In Proceedings of ACL*, pages 236–244.
- Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.
- R. Montague. 1970. Universal grammar. *Theoria*, 36(3):373–398.
- Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *NAACL-HLT*, pages 786–794.
- Bo Pang and Lillian Lee. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124.
- Francis Jeffrey Pelletier. 1994. The principle of semantic compositionality. *Topoi*, 13:11–24.
- Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of english words. In *Proceedings of ACL, ACL '93*, pages 183–190.
- Jordan B. Pollack. 1990. Recursive distributed representations. *Artificial Intelligence*, 46:77–105.
- Christian Scheible and Hinrich Schütze. 2013. Cutting recursive autoencoder trees. In *Proceedings of the International Conference on Learning Representations*.
- Hinrich Schütze. 1998. Automatic word sense discrimination. *CL*, 24(1):97–123, March.
- Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011a. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *Advances in Neural Information Processing Systems 24*.

- Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011b. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of EMNLP*, pages 151–161.
- Richard Socher, Brody Huval, Bharath Bhat, Christopher D. Manning, and Andrew Y. Ng. 2012a. Convolutional-Recursive Deep Learning for 3D Object Classification. In *Advances in Neural Information Processing Systems 25*.
- Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012b. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP-CoNLL*, pages 1201–1211.
- Mark Steedman and Jason Baldridge, 2011. *Combinatory Categorical Grammar*, pages 181–224. Wiley-Blackwell.
- Anna Szabolcsi. 1989. Bound Variables in Syntax: Are There Any? In Renate Bartsch, Johan van Benthem, and Peter van Emde Boas, editors, *Semantics and Contextual Expression*, pages 295–318. Foris, Dordrecht.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of ACL*, pages 384–394.
- Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: simple, good sentiment and topic classification. In *Proceedings of ACL*, pages 90–94.
- Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of EMNLP-HLT, HLT '05*, pages 347–354.

# Margin-based Decomposed Amortized Inference

Gourab Kundu\* and Vivek Srikumar\* and Dan Roth

University of Illinois, Urbana-Champaign

Urbana, IL. 61801

{kundu2, vsrikum2, danr}@illinois.edu

## Abstract

Given that structured output prediction is typically performed over entire datasets, one natural question is whether it is possible to re-use computation from earlier inference instances to speed up inference for future instances. Amortized inference has been proposed as a way to accomplish this. In this paper, first, we introduce a new amortized inference algorithm called the *Margin-based Amortized Inference*, which uses the notion of structured margin to identify inference problems for which previous solutions are provably optimal. Second, we introduce *decomposed amortized inference*, which is designed to address very large inference problems, where earlier amortization methods become less effective. This approach works by decomposing the output structure and applying amortization piece-wise, thus increasing the chance that we can re-use previous solutions for *parts* of the output structure. These parts are then combined to a global coherent solution using Lagrangian relaxation. In our experiments, using the NLP tasks of semantic role labeling and entity-relation extraction, we demonstrate that with the margin-based algorithm, we need to call the inference engine only for a third of the test examples. Further, we show that the decomposed variant of margin-based amortized inference achieves a greater reduction in the number of inference calls.

## 1 Introduction

A wide variety of NLP problems can be naturally cast as structured prediction problems. For

some structures like sequences or parse trees, specialized and tractable dynamic programming algorithms have proven to be very effective. However, as the structures under consideration become increasingly complex, the computational problem of predicting structures can become very expensive, and in the worst case, intractable.

In this paper, we focus on an inference technique called *amortized inference* (Srikumar et al., 2012), where previous solutions to inference problems are used to speed up new instances. The main observation that leads to amortized inference is that, very often, for different examples of the same size, the structures that maximize the score are identical. If we can efficiently identify that two inference problems have the same solution, then we can re-use previously computed structures for newer examples, thus giving us a speedup.

This paper has two contributions. First, we describe a novel algorithm for amortized inference called *margin-based amortization*. This algorithm is on an examination of the structured margin of a prediction. For a new inference problem, if this margin is larger than the sum of the decrease in the score of the previous prediction *and* any increase in the score of the second best one, then the previous solution will be the highest scoring one for the new problem. We formalize this intuition to derive an algorithm that finds provably optimal solutions and show that this approach is a generalization of previously identified schemes (based on Theorem 1 of (Srikumar et al., 2012)).

Second, we argue that the idea of amortization is best exploited at the level of *parts* of the structures rather than the entire structure because we expect a much higher redundancy in the parts. We introduce the notion of *decomposed amortized inference*, whereby we can attain a significant improvement in speedup by considering repeated sub-structures across the dataset and applying any amortized inference algorithm for the parts.

---

\* These authors contributed equally to this work.

We evaluate the two schemes and their combination on two NLP tasks where the output is encoded as a structure: PropBank semantic role labeling (Punyakanok et al., 2008) and the problem of recognizing entities and relations in text (Roth and Yih, 2007; Kate and Mooney, 2010). In these problems, the inference problem has been framed as an integer linear program (ILP). We compare our methods with previous amortized inference methods and show that margin-based amortization combined with decomposition significantly outperforms existing methods.

## 2 Problem Definition and Notation

Structured output prediction encompasses a wide variety of NLP problems like part-of-speech tagging, parsing and machine translation. The language of 0-1 integer linear programs (ILP) provides a convenient analytical tool for representing structured prediction problems. The general setting consists of binary inference variables each of which is associated with a score. The goal of inference is to find the highest scoring global assignment of the variables from a feasible set of assignments, which is defined by linear inequalities.

While efficient inference algorithms exist for special families of structures (like linear chains and trees), in the general case, inference can be computationally intractable. One approach to deal with the computational complexity of inference is to use an off-the-shelf ILP solver for solving the inference problem. This approach has seen increasing use in the NLP community over the last several years (for example, (Roth and Yih, 2004; Clarke and Lapata, 2006; Riedel and Clarke, 2006) and many others). Other approaches for solving inference include the use of cutting plane inference (Riedel, 2009), dual decomposition (Koo et al., 2010; Rush et al., 2010) and the related method of Lagrangian relaxation (Rush and Collins, 2011; Chang and Collins, 2011).

(Srikumar et al., 2012) introduced the notion of an *amortized inference algorithm*, defined as an inference algorithm that can use previous predictions to speed up inference time, thereby giving an amortized gain in inference time over the lifetime of the program.

The motivation for amortized inference comes from the observation that though the number of possible structures could be large, in practice, only a small number of these are ever seen in real

data. Furthermore, among the observed structures, a small subset typically occurs much more frequently than the others. Figure 1 illustrates this observation in the context of part-of-speech tagging. If we can efficiently characterize and identify inference instances that have the same solution, we can take advantage of previously performed computation without paying the high computational cost of inference.

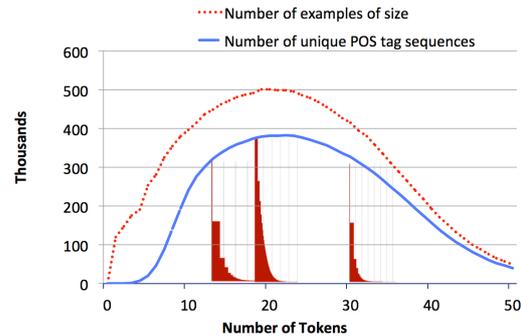


Figure 1: Comparison of number of instances and the number of unique observed part-of-speech structures in the Gigaword corpus. Note that the number of observed structures (blue solid line) is much lower than the number of sentences (red dotted line) for all sentence lengths, with the difference being very pronounced for shorter sentences. Embedded in the graph are three histograms that show the distribution of observed structures for sentences of length 15, 20 and 30. In all cases, we see that a small number of tag sequences are much more frequent than the others.

We denote inference problems by the bold-faced letters  $\mathbf{p}$  and  $\mathbf{q}$ . For a problem  $\mathbf{p}$ , the goal of inference is to jointly assign values to the parts of the structure, which are represented by a collection of inference variables  $\mathbf{y} \in \{0, 1\}^n$ . For all vectors, subscripts represent their  $i^{\text{th}}$  component.

Each  $y_i$  is associated with a real valued  $c_{\mathbf{p},i} \in \mathbb{R}$  which is the score for the variable  $y_i$  being assigned the value 1. We denote the vector comprising of all the  $c_{\mathbf{p},i}$  as  $\mathbf{c}_{\mathbf{p}}$ . The search space for assignments is restricted via constraints, which can be written as a collection of linear inequalities,  $\mathbf{M}^T \mathbf{y} \leq \mathbf{b}$ . For a problem  $\mathbf{p}$ , we denote this feasible set of structures by  $K_{\mathbf{p}}$ .

The inference problem is that of finding the feasible assignment to the structure which maximizes the dot product  $\mathbf{c}^T \mathbf{y}$ . Thus, the prediction problem can be written as

$$\arg \max_{\mathbf{y} \in K_{\mathbf{p}}} \mathbf{c}^T \mathbf{y}. \quad (1)$$

We denote the solution of this maximization problem as  $\mathbf{y}_p$ .

Let the set  $P = \{\mathbf{p}^1, \mathbf{p}^2, \dots\}$  denote previously solved inference problems, along with their respective solutions  $\{\mathbf{y}_p^1, \mathbf{y}_p^2, \dots\}$ . An *equivalence class* of integer linear programs, denoted by  $[P]$ , consists of ILPs which have the same number of inference variables and the same feasible set. Let  $K_{[P]}$  denote the feasible set of an equivalence class  $[P]$ . For a program  $\mathbf{p}$ , the notation  $\mathbf{p} \sim [P]$  indicates that it belongs to the equivalence class  $[P]$ .

(Srikumar et al., 2012) introduced a set of amortized inference schemes, each of which provides a condition for a new ILP to have the same solution as a previously seen problem. We will briefly review one exact inference scheme introduced in that work. Suppose  $\mathbf{q}$  belongs to the same equivalence class of ILPs as  $\mathbf{p}$ . Then the solution to  $\mathbf{q}$  will be the same as that of  $\mathbf{p}$  if the following condition holds for all inference variables:

$$(2\mathbf{y}_{p,i} - 1)(\mathbf{c}_{q,i} - \mathbf{c}_{p,i}) \geq 0. \quad (2)$$

This condition, referred to as *Theorem 1* in that work, is the baseline for our experiments.

In general, for any amortization scheme  $A$ , we can define two primitive operators  $\text{TESTCONDITION}_A$  and  $\text{SOLUTION}_A$ . Given a collection of previously solved problems  $P$  and a new inference problem  $\mathbf{q}$ ,  $\text{TESTCONDITION}_A(P, \mathbf{q})$  checks if the solution of the new problem is the same as that of some previously solved one and if so,  $\text{SOLUTION}_A(P, \mathbf{q})$  returns the solution.

### 3 Margin-based Amortization

In this section, we will introduce a new method for amortizing inference costs over time. The key observation that leads to this theorem stems from the *structured margin*  $\delta$  for an inference problem  $\mathbf{p} \sim [P]$ , which is defined as follows:

$$\delta = \min_{\mathbf{y} \in K_{[P]}, \mathbf{y} \neq \mathbf{y}_p} \mathbf{c}_p^T (\mathbf{y}_p - \mathbf{y}). \quad (3)$$

That is, for all feasible  $\mathbf{y}$ , we have  $\mathbf{c}_p^T \mathbf{y}_p \geq \mathbf{c}_p^T \mathbf{y} + \delta$ . The margin  $\delta$  is the upper limit on the *change in objective* that is allowed for the constraint set  $K_{[P]}$  for which the solution will not change.

For a new inference problem  $\mathbf{q} \sim [P]$ , we define  $\Delta$  as the maximum change in objective value that can be effected by an assignment that is *not* the

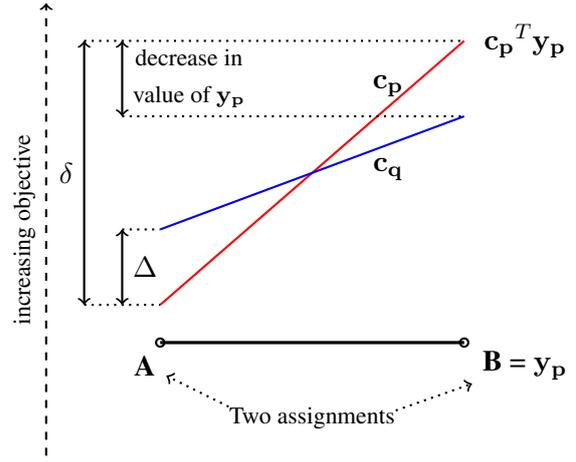


Figure 2: An illustration of the margin-based amortization scheme showing the very simple case with only two competing assignments  $\mathbf{A}$  and  $\mathbf{B}$ . Suppose  $\mathbf{B}$  is the solution  $\mathbf{y}_p$  for the inference problem  $\mathbf{p}$  with coefficients  $\mathbf{c}_p$ , denoted by the red hyperplane, and  $\mathbf{A}$  is the second-best assignment. For a new coefficient vector  $\mathbf{c}_q$ , if the margin  $\delta$  is greater than the sum of the decrease in the objective value of  $\mathbf{y}_p$  and the maximum increase in the objective of another solution ( $\Delta$ ), then the solution to the new inference problem will still be  $\mathbf{y}_p$ . The margin-based amortization theorem captures this intuition.

solution. That is,

$$\Delta = \max_{\mathbf{y} \in K_{[P]}, \mathbf{y} \neq \mathbf{y}_p} (\mathbf{c}_q - \mathbf{c}_p)^T \mathbf{y} \quad (4)$$

Before stating the theorem, we will provide an intuitive explanation for it. Moving from  $\mathbf{c}_p$  to  $\mathbf{c}_q$ , consider the sum of the decrease in the value of the objective for the solution  $\mathbf{y}_p$  and  $\Delta$ , the maximum change in objective value for an assignment that is not the solution. If this sum is less than the margin  $\delta$ , then no other solution will have an objective value higher than  $\mathbf{y}_p$ . Figure 2 illustrates this using a simple example where there are only two competing solutions.

This intuition is captured by our main theorem which provides a condition for problems  $\mathbf{p}$  and  $\mathbf{q}$  to have the same solution  $\mathbf{y}_p$ .

**Theorem 1 (Margin-based Amortization).** *Let  $\mathbf{p}$  denote an inference problem posed as an integer linear program belonging to an equivalence class  $[P]$  with optimal solution  $\mathbf{y}_p$ . Let  $\mathbf{p}$  have a structured margin  $\delta$ , i.e., for any  $\mathbf{y}$ , we have  $\mathbf{c}_p^T \mathbf{y}_p \geq \mathbf{c}_p^T \mathbf{y} + \delta$ . Let  $\mathbf{q} \sim [P]$  be another inference instance in the same equivalence class and let  $\Delta$  be defined as in Equation 4. Then,  $\mathbf{y}_p$  is the solution of the problem  $\mathbf{q}$  if the following holds:*

$$-(\mathbf{c}_q - \mathbf{c}_p)^T \mathbf{y}_p + \Delta \leq \delta \quad (5)$$

*Proof.* For some feasible  $\mathbf{y}$ , we have

$$\begin{aligned} \mathbf{c}_q^T \mathbf{y}_p - \mathbf{c}_q^T \mathbf{y} &\geq \mathbf{c}_q^T \mathbf{y}_p - \mathbf{c}_p^T \mathbf{y} - \Delta \\ &\geq \mathbf{c}_q^T \mathbf{y}_p - \mathbf{c}_p^T \mathbf{y}_p + \delta - \Delta \\ &\geq 0 \end{aligned}$$

The first inequality comes from the definition of  $\Delta$  in (4) and the second one follows from the definition of  $\delta$ . The condition of the theorem in (5) gives us the final step. For any feasible  $\mathbf{y}$ , the objective score assigned to  $\mathbf{y}_p$  is greater than the score assigned to  $\mathbf{y}$  according to problem  $\mathbf{q}$ . That is,  $\mathbf{y}_p$  is the solution to the new problem.  $\square$

The margin-based amortization theorem provides a general, new amortized inference algorithm. Given a new inference problem, we check whether the inequality (5) holds for any previously seen problems in the same equivalence class. If so, we return the cached solution. If no such problem exists, then we make a call to an ILP solver.

Even though the theorem provides a condition for two integer linear programs to have the same solution, checking the validity of the condition requires the computation of  $\Delta$ , which in itself is another integer linear program. To get around this, we observe that if any constraints in Equation 4 are relaxed, the value of the resulting maximum can only increase. Even with the increased  $\Delta$ , if the condition of the theorem holds, then the rest of the proof follows and hence the new problem will have the same solution. In other words, we can solve relaxed, tractable variants of the maximization in Equation 4 and still retain the guarantees provided by the theorem. The tradeoff is that, by doing so, the condition of the theorem will apply to fewer examples than theoretically possible. In our experiments, we will define the relaxation for each problem individually and even with the relaxations, the inference algorithm based on the margin-based amortization theorem outperforms all previous amortized inference algorithms.

The condition in inequality (5) is, in fact, a strict generalization of the condition for Theorem 1 in (Srikumar et al., 2012), stated in (2). If the latter condition holds, then we can show that  $\Delta \leq 0$  and  $(\mathbf{c}_q - \mathbf{c}_p)^T \mathbf{y}_p \geq 0$ . Since  $\delta$  is, by definition, non-negative, the margin-based condition is satisfied.

## 4 Decomposed Amortized Inference

One limitation in previously considered approaches for amortized inference stems from the

expectation that the same *full* assignment maximizes the objective score for different inference problems, or equivalently, that the entire structure is repeated multiple times. Even with this assumption, we observe a speedup in prediction.

However, intuitively, even if entire structures are not repeated, we expect parts of the assignment to be the same across different instances. In this section, we address the following question: *Can we take advantage of the redundancy in components of structures to extend amortization techniques to cases where the full structured output is not repeated?* By doing so, we can store partial computation for future inference problems.

For example, consider the task of part of speech tagging. While the likelihood of two long sentences having the same part of speech tag sequence is not high, it is much more likely that shorter sections of the sentences will share the same tag sequence. We see from Figure 1 that the number of possible structures for shorter sentences is much smaller than the number of sentences. This implies that many shorter sentences share the same structure, thus improving the performance of an amortized inference scheme for such inputs. The goal of decomposed amortized inference is to extend this improvement to larger problems by increasing the size of equivalence classes.

To decompose an inference problem, we use the approach of Lagrangian Relaxation (Lemaréchal, 2001) that has been used successfully for various NLP tasks (Chang and Collins, 2011; Rush and Collins, 2011). We will briefly review the underlying idea<sup>1</sup>. The goal is to solve an integer linear program  $\mathbf{q}$ , which is defined as

$$\mathbf{q} : \max_{\mathbf{M}^T \mathbf{y} \leq \mathbf{b}} \mathbf{c}_q^T \mathbf{y}$$

We partition the constraints into two sets, say  $C_1$  denoting  $\mathbf{M}_1^T \mathbf{y} \leq \mathbf{b}_1$  and  $C_2$ , denoting constraints  $\mathbf{M}_2^T \mathbf{y} \leq \mathbf{b}_2$ . The assumption is that in the absence the constraints  $C_2$ , the inference problem becomes computationally easier to solve. In other words, we can assume the existence of a subroutine that can efficiently compute the solution of the relaxed problem  $\mathbf{q}'$ :

$$\mathbf{q}' : \max_{\mathbf{M}_1^T \mathbf{y} \leq \mathbf{b}_1} \mathbf{c}_q^T \mathbf{y}$$

<sup>1</sup>For simplicity, we only write inequality constraints in the paper. However, all the results here are easily extensible to equality constraints by removing the non-negativity constraints from the corresponding dual variables.

We define Lagrange multipliers  $\Lambda \geq \mathbf{0}$ , with one  $\lambda_i$  for each constraint in  $C_2$ . For problem  $\mathbf{q}$ , we can define the Lagrangian as

$$L(\mathbf{y}, \Lambda) = \mathbf{c}_{\mathbf{q}}^T \mathbf{y} - \Lambda^T (\mathbf{M}_2^T \mathbf{y} - \mathbf{b}_2)$$

Here, the domain of  $\mathbf{y}$  is specified by the constraint set  $C_1$ . The dual objective is

$$\begin{aligned} L(\Lambda) &= \max_{\mathbf{M}_1^T \mathbf{y} \leq \mathbf{b}_1} \mathbf{c}_{\mathbf{q}}^T \mathbf{y} - \Lambda^T (\mathbf{M}_2^T \mathbf{y} - \mathbf{b}_2) \\ &= \max_{\mathbf{M}_1^T \mathbf{y} \leq \mathbf{b}_1} (\mathbf{c}_{\mathbf{q}} - \Lambda^T \mathbf{M}_2)^T \mathbf{y} + \Lambda^T \mathbf{b}_2. \end{aligned}$$

Note that the maximization in the definition of the dual objective has the same functional form as  $\mathbf{q}'$  and any approach to solve  $\mathbf{q}'$  can be used here to find the dual objective  $L(\Lambda)$ . The dual of the problem  $\mathbf{q}$ , given by  $\min_{\Lambda \geq \mathbf{0}} L(\Lambda)$ , can be solved using subgradient descent over the dual variables.

Relaxing the constraints  $C_2$  to define the problem  $\mathbf{q}'$  has several effects. First, it can make the resulting inference problem  $\mathbf{q}'$  easier to solve. More importantly, removing constraints can also lead to the merging of multiple equivalence classes, leading to fewer, more populous equivalence classes. Finally, removing constraints can decompose the inference problem  $\mathbf{q}'$  into smaller independent sub-problems  $\{\mathbf{q}^1, \mathbf{q}^2, \dots\}$  such that no constraint that is in  $C_1$  has active variables from two different sets in the partition.

For the sub-problem  $\mathbf{q}^i$  comprising of variables  $\mathbf{y}^i$ , let the corresponding objective coefficients be  $\mathbf{c}_{\mathbf{q}^i}$  and the corresponding sub-matrix of  $\mathbf{M}_2$  be  $\mathbf{M}_2^i$ . Now, we can define the dual-augmented sub-problem as

$$\max_{\mathbf{M}_1^T \mathbf{y} \leq \mathbf{b}_1} (\mathbf{c}_{\mathbf{q}^i} - \Lambda^T \mathbf{M}_2^i)^T \mathbf{y}^i \quad (6)$$

Solving all such sub-problems will give us a complete assignment for all the output variables.

We can now define the decomposed amortized inference algorithm (Algorithm 1) that performs sub-gradient descent over the dual variables. The input to the algorithm is a collection of previously solved problems with their solutions, a new inference problem  $\mathbf{q}$  and an amortized inference scheme  $A$  (such as the margin-based amortization scheme). In addition, for the task at hand, we first need to identify the set of constraints  $C_2$  that can be introduced via the Lagrangian.

First, we check if the solution can be obtained without decomposition (lines 1–2). Otherwise,

---

### Algorithm 1 Decomposed Amortized Inference

---

**Input:** A collection of previously solved inference problems  $P$ , a new problem  $\mathbf{q}$ , an amortized inference algorithm  $A$ .

**Output:** The solution to problem  $\mathbf{q}$

```

1: if TESTCONDITION( $A, \mathbf{q}, P$ ) then
2:   return SOLUTION( $A, \mathbf{q}, P$ )
3: else
4:   Initialize  $\lambda_i \leftarrow 0$  for each constraint in  $C_2$ .
5:   for  $t = 1 \dots T$  do
6:     Partition the problem  $\mathbf{q}$  into sub-
       problems  $\mathbf{q}^1, \mathbf{q}^2, \dots$  such that no con-
       straint in  $C_1$  has active variables from
       two partitions.
7:     for partition  $\mathbf{q}^i$  do
8:        $\mathbf{y}^i \leftarrow$  Solve the maximization prob-
       lem for  $\mathbf{q}^i$  (Eq. 6) using the amortized
       scheme  $A$ .
9:     end for
10:    Let  $\mathbf{y} \leftarrow [\mathbf{y}^1; \mathbf{y}^2; \dots]$ 
11:    if  $\mathbf{M}_2 \mathbf{y} \leq \mathbf{b}_2$  and  $(\mathbf{b}_2 - \mathbf{M}_2 \mathbf{y})_i \lambda_i = 0$ 
       then
12:      return  $\mathbf{y}$ 
13:    else
14:       $\Lambda \leftarrow [\Lambda - \mu_t (\mathbf{b}_2 - \mathbf{M}_2^T \mathbf{y})]_+$ 
15:    end if
16:  end for
17:  return solution of  $\mathbf{q}$  using a standard infer-
       ence algorithm
18: end if

```

---

we initialize the dual variables  $\Lambda$  and try to obtain the solution iteratively. At the  $t^{\text{th}}$  iteration, we partition the problem  $\mathbf{q}$  into sub-problems  $\{\mathbf{q}^1, \mathbf{q}^2, \dots\}$  as described earlier (line 6). Each partition defines a smaller inference problem with its own objective coefficients and constraints. We can apply the amortization scheme  $A$  to each sub-problem to obtain a complete solution for the relaxed problem (lines 7–10). If this solution satisfies the constraints  $C_2$  and complementary slackness conditions, then the solution is provably the maximum of the problem  $\mathbf{q}$ . Otherwise, we take a subgradient step to update the value of  $\Lambda$  using a step-size  $\mu_t$ , subject to the constraint that all dual variables must be non-negative (line 14). If we do not converge to a solution in  $T$  iterations, we call the underlying solver on the full problem.

In line 8 of the algorithm, we make multiple calls to the underlying amortized inference procedure to solve each sub-problem. If the sub-

problem cannot be solved using the procedure, then we can either solve the sub-problem using a different approach (effectively giving us the standard Lagrangian relaxation algorithm for inference), or we can treat the full instance as a cache miss and make a call to an ILP solver. In our experiments, we choose the latter strategy.

## 5 Experiments and Results

Our experiments show two results: 1. The margin-based scheme outperforms the amortized inference approaches from (Srikumar et al., 2012). 2. Decomposed amortized inference gives further gains in terms of re-using previous solutions.

### 5.1 Tasks

We report the performance of inference on two NLP tasks: semantic role labeling and the task of extracting entities and relations from text. In both cases, we used an existing formulation for structured inference and only modified the inference calls. We will briefly describe the problems and the implementation and point the reader to the literature for further details.

**Semantic Role Labeling (SRL)** Our first task is that of identifying arguments of verbs in a sentence and annotating them with semantic roles (Gildea and Jurafsky, 2002; Palmer et al., 2010). For example, in the sentence *Mrs. Haag plays Eltiani.*, the verb *plays* takes two arguments: *Mrs. Haag*, the actor, labeled as A0 and *Eltiani*, the role, labeled as A1. It has been shown in prior work (Punyakank et al., 2008; Toutanova et al., 2008) that making a globally coherent prediction boosts performance of SRL.

In this work, we used the SRL system of (Punyakank et al., 2008), where one inference problem is generated for each verb and each inference variable encodes the decision that a given constituent in the sentence takes a specific role. The scores for the inference variables are obtained from a classifier trained on the PropBank corpus. Constraints encode structural and linguistic knowledge about the problem. For details about the formulations of the inference problem, please see (Punyakank et al., 2008).

Recall from Section 3 that we need to define a relaxed version of the inference problem to efficiently compute  $\Delta$  for the margin-based approach. For a problem instance with coefficients  $c_q$  and cached coefficients  $c_p$ , we take the sum of the

highest  $n$  values of  $c_q - c_p$  as our  $\Delta$ , where  $n$  is the number of argument candidates to be labeled.

To identify constraints that can be relaxed for the decomposed algorithm, we observe that most constraints are not predicate specific and apply for all predicates. The only constraint that is predicate specific requires that each predicate can only accept roles from a list of roles that is defined for that predicate. By relaxing this constraint in the decomposed algorithm, we effectively merge all the equivalence classes for all predicates with a specific number of argument candidates.

**Entity-Relation extraction** Our second task is that of identifying the types of entities in a sentence and the relations among them, which has been studied by (Roth and Yih, 2007; Kate and Mooney, 2010) and others. For the sentence *Oswald killed Kennedy*, the words *Oswald* and *Kennedy* will be labeled by the type PERSON, and the KILL relation exists between them.

We followed the experimental setup as described in (Roth and Yih, 2007). We defined one inference problem for each sentence. For every entity (which is identified by a constituent in the sentence), an inference variable is introduced for each entity type. For each pair of constituents, an inference variable is introduced for each relation type. Clearly, the assignment of types to entities and relations are not independent. For example, an entity of type ORGANIZATION cannot participate in a relation of type BORN-IN because this relation label can connect entities of type PERSON and LOCATION only. Incorporating these natural constraints during inference were shown to improve performance significantly in (Roth and Yih, 2007). We trained independent classifiers for entities and relations and framed the inference problem as in (Roth and Yih, 2007). For further details, we refer the reader to that paper.

To compute the value of  $\Delta$  for the margin-based algorithm, for a new instance with coefficients  $c_q$  and cached coefficients  $c_p$ , we define  $\Delta$  to be the sum of all non-negative values of  $c_q - c_p$ .

For the decomposed inference algorithm, if the number of entities is less than 5, no decomposition is performed. Otherwise, the entities are partitioned into two sets: set  $A$  includes the first four entities and set  $B$  includes the rest of the entities. We relaxed the relation constraints that go across these two sets of entities to obtain two independent inference problems.

## 5.2 Experimental Setup

We follow the experimental setup of (Srikumar et al., 2012) and simulate a long-running NLP process by caching problems and solutions from the Gigaword corpus. We used a database engine to cache ILP and their solutions along with identifiers for the equivalence class and the value of  $\delta$ .

For the margin-based algorithm and the Theorem 1 from (Srikumar et al., 2012), for a new inference problem  $\mathbf{p} \sim [P]$ , we retrieve all inference problems from the database that belong to the same equivalence class  $[P]$  as the test problem  $\mathbf{p}$  and find the cached assignment  $\mathbf{y}$  that has the highest score according to the coefficients of  $\mathbf{p}$ . We only consider cached ILPs whose solution is  $\mathbf{y}$  for checking the conditions of the theorem. This optimization ensures that we only process a small number of cached coefficient vectors.

In a second efficiency optimization, we pruned the database to remove redundant inference problems. A problem is redundant if solution to that problem can be inferred from the other problems stored in the database that have the same solution and belong to the same equivalence class. However, this pruning can be computationally expensive if the number of problems with the same solution and the same equivalence class is very large. In that case, we first sampled a 5000 problems randomly and selected the non-redundant problems from this set to keep in the database.

## 5.3 Results

We compare our approach to a state-of-the-art ILP solver<sup>2</sup> and also to Theorem 1 from (Srikumar et al., 2012). We choose this baseline because it is shown to give the highest improvement in wall-clock time and also in terms of the number of cache hits. However, we note that the results presented in our work outperform all the previous amortization algorithms, including the approximate inference methods.

We report two performance metrics – the percentage decrease in the number of ILP calls, and the percentage decrease in the wall-clock inference time. These are comparable to the *speedup* and *clock speedup* defined in (Srikumar et al., 2012). For measuring time, since other aspects of prediction (like feature extraction) are the same across all settings, we only measure the time taken for inference and ignore other aspects. For both

<sup>2</sup>We used the Gurobi optimizer for our experiments.

tasks, we report the runtime performance on section 23 of the Penn Treebank. Note that our amortization schemes *guarantee optimal solution*. Consequently, using amortization, task accuracy remains the same as using the original solver.

Table 1 shows the percentage reduction in the number of calls to the ILP solver. Note that for both the SRL and entity-relation problems, the margin-based approach, even without using decomposition (the columns labeled **Original**), outperforms the previous work. Applying the decomposed inference algorithm improves both the baseline and the margin-based approach. Overall, however, the fewest number of calls to the solver is made when combining the decomposed inference algorithm with the margin-based scheme. For the semantic role labeling task, we need to call the solver only for one in six examples while for the entity-relations task, only one in four examples require a solver call.

Table 2 shows the corresponding reduction in the wall-clock time for the various settings. We see that once again, the margin based approach outperforms the baseline. While the decomposed inference algorithm improves running time for SRL, it leads to a slight increase for the entity-relation problem. Since this increase occurs in spite of a reduction in the number of solver calls, we believe that this aspect can be further improved with an efficient implementation of the decomposed inference algorithm.

## 6 Discussion

**Lagrangian Relaxation in the literature** In the literature, in applications of the Lagrangian relaxation technique (such as (Rush and Collins, 2011; Chang and Collins, 2011; Reichart and Barzilay, 2012) and others), the relaxed problems are solved using specialized algorithms. However, in both the relaxations considered in this paper, even the relaxed problems cannot be solved without an ILP solver, and yet we can see improvements from decomposition in Table 1.

To study the impact of amortization on running time, we modified our decomposition based inference algorithm to solve each sub-problem using the ILP solver instead of amortization. In these experiments, we ran Lagrangian relaxation for until convergence or at most  $T$  iterations. After  $T$  iterations, we call the ILP solver and solve the original problem. We set  $T$  to 100 in one set of exper-

Method	% ILP Solver calls required			
	Semantic Role Labeling		Entity-Relation Extraction	
	Original	+ Decomp.	Original	+ Decomp.
ILP Solver	100	–	100	–
(Srikumar et al., 2012)	41	24.4	59.5	57.0
Margin-based	32.7	16.6	28.2	25.4

Table 1: Reduction in number of inference calls

Method	% time required compared to ILP Solver			
	Semantic Role Labeling		Entity-Relation Extraction	
	Original	+ Decomp.	Original	+ Decomp.
ILP Solver	100	–	100	–
(Srikumar et al., 2012)	54.8	40.0	81	86
Margin-based	45.9	38.1	58.1	61.3

Table 2: Reduction in inference time

iments (call it  $Lag1$ ) and  $T$  to 1 (call it  $Lag2$ ). In SRL, compared to solving the original problem with ILP Solver, both  $Lag1$  and  $Lag2$  are roughly 2 times slower. For entity relation task, compared to ILP Solver,  $Lag1$  is 186 times slower and  $Lag2$  is 1.91 times slower. Since we used the same implementation of the decomposition in all experiments, this shows that the decomposed inference algorithm crucially benefits from the underlying amortization scheme.

**Decomposed amortized inference** The decomposed amortized inference algorithm helps improve amortized inference in two ways. First, since the number of structures is a function of its size, considering smaller sub-structures will allow us to cache inference problems that cover a larger subset of the space of possible sub-structures. We observed this effect in the problem of extracting entities and relations in text. Second, removing a constraint need not always partition the structure into a set of smaller structures. Instead, by removing the constraint, examples that might have otherwise been in different equivalence classes become part of a combined, larger equivalence class. Increasing the size of the equivalence classes increases the probability of a cache-hit. In our experiments, we observed this effect in the SRL task.

## 7 Conclusion

Amortized inference takes advantage of the regularities in structured output to re-use previous computation and improve running time over the lifetime of a structured output predictor. In this paper, we have described two approaches for amortizing inference costs over datasets. The first, called the *margin-based amortized inference*, is a

new, provably exact inference algorithm that uses the notion of a structured margin to identify previously solved problems whose solutions can be re-used. The second, called *decomposed amortized inference*, is a meta-algorithm over any amortized inference that takes advantage of previously computed sub-structures to provide further reductions in the number of inference calls. We show via experiments that these methods individually give a reduction in the number of calls made to an inference engine for semantic role labeling and entity-relation extraction. Furthermore, these approaches complement each other and, together give an additional significant improvement.

## Acknowledgments

The authors thank the members of the Cognitive Computation Group at the University of Illinois for insightful discussions and the anonymous reviewers for valuable feedback. This research is sponsored by the Army Research Laboratory (ARL) under agreement W911NF-09-2-0053. The authors also gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. This material also is based on research sponsored by DARPA under agreement number FA8750-13-2-0008. This work has also been supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20155. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of ARL, DARPA, AFRL, IARPA, DoI/NBC or the US government.

## References

- Y-W. Chang and M. Collins. 2011. Exact decoding of phrase-based translation models through Lagrangian relaxation. *EMNLP*.
- J. Clarke and M. Lapata. 2006. Constraint-based sentence compression: An integer programming approach. In *ACL*.
- D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*.
- R. Kate and R. Mooney. 2010. Joint entity and relation extraction using card-pyramid parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 203–212. Association for Computational Linguistics.
- T. Koo, A. M. Rush, M. Collins, T. Jaakkola, and D. Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *EMNLP*.
- C. Lemaréchal. 2001. Lagrangian Relaxation. In *Computational Combinatorial Optimization*, pages 112–156.
- M. Palmer, D. Gildea, and N. Xue. 2010. *Semantic Role Labeling*, volume 3. Morgan & Claypool Publishers.
- V. Punyakanok, D. Roth, and W. Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*.
- R. Reichart and R. Barzilay. 2012. Multi event extraction guided by global constraints. In *NAACL*, pages 70–79.
- S. Riedel and J. Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *EMNLP*.
- S. Riedel. 2009. Cutting plane MAP inference for Markov logic. *Machine Learning*.
- D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *CoNLL*.
- D. Roth and W. Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*.
- A.M. Rush and M. Collins. 2011. Exact decoding of syntactic translation models through Lagrangian relaxation. In *ACL*, pages 72–82, Portland, Oregon, USA, June.
- A. M. Rush, D. Sontag, M. Collins, and T. Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *EMNLP*.
- V. Srikumar, G. Kundu, and D. Roth. 2012. On amortizing inference cost for structured prediction. In *EMNLP*.
- K. Toutanova, A. Haghghi, and C. D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34:161–191.

# Semi-Supervised Semantic Tagging of Conversational Understanding using Markov Topic Regression

<b>Asli Celikyilmaz</b> Microsoft Mountain View, CA, USA asli@ieee.org	<b>Dilek Hakkani-Tur, Gokhan Tur</b> Microsoft Research Mountain View, CA, USA dilek@ieee.org gokhan.tur@ieee.org	<b>Ruhi Sarikaya</b> Microsoft Redmond, WA, USA rusarika@microsoft.com
---	---	---

## Abstract

Finding concepts in natural language utterances is a challenging task, especially given the scarcity of labeled data for learning semantic ambiguity. Furthermore, data mismatch issues, which arise when the expected test (target) data does not exactly match the training data, aggravate this scarcity problem. To deal with these issues, we describe an efficient semi-supervised learning (SSL) approach which has two components: (i) *Markov Topic Regression* is a new probabilistic model to cluster words into semantic tags (concepts). It can efficiently handle semantic ambiguity by extending standard topic models with two new features. First, it encodes word n-gram features from labeled source and unlabeled target data. Second, by going beyond a bag-of-words approach, it takes into account the *inherent sequential nature* of utterances to learn semantic classes based on context. (ii) *Retrospective Learner* is a new learning technique that adapts to the unlabeled target data. Our new SSL approach improves semantic tagging performance by 3% absolute over the baseline models, and also compares favorably on semi-supervised syntactic tagging.

## 1 Introduction

Semantic tagging is used in natural language understanding (NLU) to recognize words of semantic importance in an utterance, such as entities. Typically, a semantic tagging model requires large amount of domain specific data to achieve good

performance (Tur and DeMori, 2011). This requires a tedious and time intensive data collection and labeling process. In the absence of large labeled training data, the tagging model can behave poorly on test data (target domain). This is usually caused by data mismatch issues and lack of coverage that arise when the target data does not match the training data.

To deal with these issues, we present a new semi-supervised learning (SSL) approach, which mainly has two components. It initially starts with training supervised Conditional Random Fields (CRF) (Lafferty et al., 2001) on the source training data which has been semantically tagged. Using the trained model, it decodes unlabeled dataset from the target domain. With the data mismatch issues in mind, to correct errors that the supervised model make on the target data, the SSL model leverages the additional information by way of a new clustering method. Our first contribution is a new probabilistic topic model, *Markov Topic Regression* (MTR), which uses rich features to capture the degree of association between words and semantic tags. First, it encodes the n-gram context features from the labeled source data and the unlabeled target data as prior information to learn semantic classes based on context. Thus, each latent semantic class corresponds to one of the semantic tags found in labeled data. MTR is not invariant to reshuffling of words due to its Markovian property; hence, word-topic assignments are also affected by the topics of the surrounding words. Because of these properties, MTR is less sensitive to the errors caused by the semantic ambiguities. Our SSL uses MTR to smooth the semantic tag posteriors on the unlabeled target data (decoded using the CRF model) and later obtains the best tag sequences. Using the labeled source and automati-

cally labeled target data, it re-trains a new CRF-model.

Although our iterative SSL learning model can deal with the training and test data mismatch, it neglects the performance effects caused by adapting the source domain to the target domain. In fact, most SSL methods used for adaptation, e.g., (Zhu, 2005), (Daumé-III, 2010), (Subramanya et al., 2010), etc., do not emphasize this issue. With this in mind, we introduce a new iterative training algorithm, *Retrospective Learning*, as our second contribution. While retrospective learning iteratively trains CRF models with the automatically annotated target data (explained above), it keeps track of the errors of the previous iterations so as to carry the properties of both the source and target domains.

In short, through a series of experiments we show how MTR clustering provides additional information to SSL on the target domain utterances, and greatly impacts semantic tagging performance. Specifically, we analyze MTR’s performance on two different types of semantic tags: named-entities and descriptive tags as shown in Table 1. Our experiments show that it is much harder to detect descriptive tags compared to named-entities.

Our SSL approach uses probabilistic clustering method tailored for tagging natural language utterances. To the best of our knowledge, our work is the first to explore the unlabeled data to iteratively adapt the semantic tagging models for target domains, preserving information from the previous iterations. With the hope of spurring related work in domains such as entity detection, syntactic tagging, etc., we extend the earlier work on SSL part-of-speech (POS) tagging and show in the experiments that our approach is not only useful for semantic tagging but also syntactic tagging.

The remainder of this paper is divided as follows: §2 gives background on SSL and semantic clustering methods, §3 describes our new clustering approach, §4 presents the new iterative learning, §5 presents our experimental results and §6 concludes our paper.

## 2 Related Work and Motivation

**(I) Semi-Supervised Tagging.** Supervised methods for semantic tagging in NLU require a large number of in-domain human-labeled utterances and gazetteers (movie, actor names, etc.), increas-

<ul style="list-style-type: none"> <li>• Are there any [<i>comedies</i>] with [<i>Ryan Gosling</i>]?</li> <li>• How about [<i>oscar winning</i>] movies by [<i>James Cameron</i>]?</li> <li>• Find [<i>Woody Allen</i>] movies similar to [<i>Manhattan</i>].</li> </ul> <p>[Named Entities]</p> <p>director: <i>James Cameron, Woody Allen,...</i></p> <p>actor: <i>Ryan Gosling, Woody Allen,...</i></p> <p>title: <i>Manhattan, Midnight in Paris,...</i></p> <p>[Descriptive Tags]</p> <p>restriction: <i>similar, suitable, free, rate,...</i></p> <p>description: <i>oscar winning, new release, gardening,...</i></p> <p>genre: <i>spooky, comedies, feel good, romance,...</i></p>
--

Table 1: Samples of semantically tagged utterances from movie domain, named-entities and descriptive tags.

ing the need for significant manual labor (Tur and DeMori, 2011). Recent work on similar tasks overcome these challenges using SSL methods as follows:

- (Wang et al., 2009; Li et al., 2009; Li, 2010; Liu et al., 2011) investigate web query tagging using semi-supervised sequence models. They extract semantic lexicons from unlabeled web queries, to use as features. Our work differs from these, in that, rather than just detecting named-entities, our utterances include *descriptive* tags (see Table 1).

- Typically the source domain has different distribution than the target domain, due to topic shifts in time, newly introduced features (e.g., until recently online articles did not include facebook “like” feature.), etc. Adapting the source domain using unlabeled data is the key to achieving good performance across domains. Recent adaptation methods for SSL use: expectation minimization (Daumé-III, 2010) graph-based learning (Chapelle et al., 2006; Zhu, 2005), etc. In (Subramanya et al., 2010) an efficient iterative SSL method is described for syntactic tagging, using graph-based learning to smooth POS tag posteriors. However, (Reisinger and Mooney, 2011) argues that vector space models, such as graph-learning, may fail to capture the richness of word meaning, as similarity is not a globally consistent metric. Rather than graph-learning, we present a new SSL using a probabilistic model, MTR, to cluster words based on co-occurrence statistics.

- Most iterative SSL methods, do not keep track of the errors made, nor consider the divergence from the original model. (Lavoie et al., 2011) argues that iterative learning models should mitigate new errors made by the model at each iteration by

keeping the history of the prior predictions. This ensures that a penalty is paid for diverging from the previous model’s predictions, which will be traded off against the benefit of reducing classification loss. We present a *retrospective* SSL for CRF, in that, the iterative learner keeps track of the errors of the previous iterations so as to carry the properties of both the source and target domains.

**(II) Semantic Clustering.** A common property of several context-based word clustering techniques, e.g., Brown clustering (Brown et al., 1992), Clustering by Committee (Pantel, 2003), etc., is that they mainly cluster based on local context such as nearby words. Standard topic models, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003), use a *bag-of-words* approach, which disregards word order and clusters words together that appear in a similar global context. Such models have been effective in discovering lexicons in many NLP tasks, e.g., named-entity recognition (Guo et al., 2009), word-sense disambiguation (Boyd-Graber et al., 2007; Li et al., 2010), syntactic/semantic parsing (Griffiths et al., 2005; Singh et al., 2010), speaker identification (Nyugen et al., 2012), etc. Recent topic models consider word sequence information in documents (Griffiths et al., 2005; Moon et al., 2010). The Hidden Topic Markov Model (HTMM) by (Gruber et al., 2005), for instance, models sentences in documents as Markov chains, assuming all words in a sentence have the *same* topic. While MTR has a similar Markovian property, we encode features on words to allow each word in an utterance to sample from any of the given semantic tags, as in “what are [scary]<sub>genre</sub> movies by [Hitchcock]<sub>director</sub>?”.

In LDA, common words tend to dominate all topics causing related words to end up in different topics. In (Petterson et al., 2010), the vector-based features of words are used as prior information in LDA so that the words that are synonyms end up in same topic. Thus, we build a semantically rich topic model, MTR, using *word context* features as side information. Using a smoothing prior for each word-topic pair (instead of a constant  $\beta$  smoother), MTR assures that the words are distributed over topics based on how similar they are. (e.g., “scary” and “spooky”, which have similar context features, go into the same semantic tag, “genre”). Thus, to best of our knowledge, MTR is the first topic model to incorporate word features while considering the sequence of words.

### 3 Markov Topic Regression - MTR

#### 3.1 Model and Abstractions

LDA assumes that the latent topics of documents are sampled independently from one of  $K$  topics. MTR breaks down this independence assumption by allowing Markov relations between the hidden tags to capture the relations between consecutive words (as sketched in Figure 1 and Algorithm 1).

**(I) Semantic Tags ( $s_i$ ):** Each word  $w_i$  of a given utterance with  $N_j$  words,  $u_j = \{w_i\}_{i=1}^{N_j} \in U$ ,  $j=1, \dots, |U|$ , from a set of utterances  $U$ , is associated with a latent semantic tag (state) variable  $s_i \in \mathcal{S}$ , where  $\mathcal{S}$  is the set of semantic tags. We assume a fixed  $K$  topics corresponding to semantic tags of labeled data. In a similar way to HTMM (Gruber et al., 2005) described for documents, MTR samples each  $s_i$  from a Markov chain that is specific to its utterance  $u_j$ . Each state  $s_i$  generates a word,  $w_i$ , based on the word-state co-occurrences. MTR allows for sampling of consecutive words from different tag clusters. The initial probabilities of the latent states are sampled from a Dirichlet distribution over state variables,  $\theta_j$ , with  $\alpha$  hyperparameter for each  $u_j$ .

**(II) Tag Transition Indicator ( $\psi_v$ ):** Given utterance  $u_j$ , the decision to sample a  $w_i$  from a new topic is determined by an indicator variable,  $c_{j,i}$ , that is sampled from a *Binomial*( $\psi_{v=w_i}$ ) distribution with a *Beta* conjugate prior. (There are  $v$  binomials for each vocabulary term.)  $c_{j,i}=1$  suggests that a new state be sampled from  $K$  possible tags for the word  $w_i$  in  $u_j$ , and  $c_{j,i}=0$  suggests that the state  $s_i$  of  $w_i$  should be the same as the previous word’s latent state  $s_{i-1}$ . The first position of the sequence is sampled from a new state, hence  $c_{j,i=1}=1$ .

**(III) Tag Transition Base Measure ( $\eta$ ):** Prior probability of a word given a tag should increase the chances of sampling words from the correct semantic tag. MTR constrains the generation of a tag  $s_i$  given the previous tag  $s_{i-1}$  and the current  $w_i$  based on  $c_{j,i}$  by using a vocabulary specific Beta prior,  $\psi_v \sim \text{Beta}(\eta_v)$ <sup>1</sup>, on each word in vocabulary  $w_{v=1, \dots, V}$ . We inject the prior information on semantic tags to define values of the base measure  $\eta_v$  using external knowledge from two sources:

(a) **Entity Priors ( $\eta_S$ ):** Prior probability on named-entities and descriptive tags denoted as

<sup>1</sup>For each beta distribution we use symmetric  $\text{Beta}(\eta_v) = \text{Beta}(\alpha=\eta_v, \beta=\eta_v)$ .

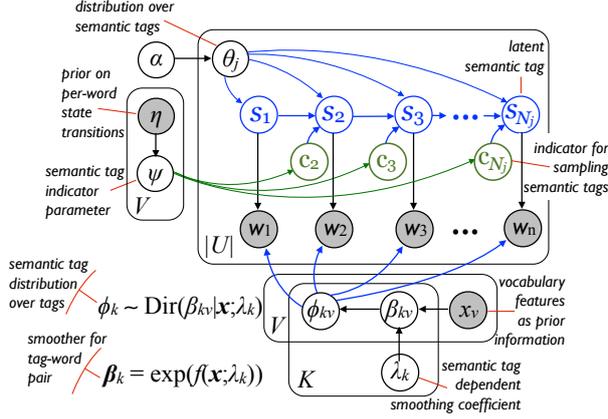


Figure 1: The graph representation of the Markov Topic Regression (MTR). To demonstrate hidden state Markov Chain, the generation of each word is explicitly shown (inside of the plate).

$\eta_S = p(s_i | s_{i-1}, w_i = v, w_{i-1})$ . We use web sources (wiki pages on movies and urls such as imdb.com) and labeled training data to extract entity lists that correspond to the semantic tags of our domains. We keep the frequency of each n-gram to convert into (empirical) prior probability distribution.

(b) **Language Model Prior ( $\eta_W$ ):** Probabilities on word transitions denoted as  $\eta_W = p(w_i = v | w_{i-1})$ . We built a language model using SRILM (Stolcke, 2002) on the domain specific sources such as top wiki pages and blogs on online movie reviews, etc., to obtain the probabilities of domain-specific n-grams, up to 3-grams. The observed priors,  $\eta_S$  and  $\eta_W$ , are used for calculating the base measure  $\eta$  for each vocabulary  $w_v$  as:

$$\eta_v^{s_i | s_{i-1}} = \begin{cases} \eta_S^{s_i | s_{i-1}, w_i = v}, & \text{if } \eta_S^{s_i | s_{i-1}, w_i = v} \text{ exists,} \\ \eta_W^{w_i = v, w_{i-1}}, & \text{otherwise} \end{cases} \quad (1)$$

In Eq.(1), we assume that the prior on the semantic tags,  $\eta_S$ , is more indicative of the decision for sampling a  $w_i$  from a new tag compared to language model posteriors on word sequences,  $\eta_W$ . Here we represent the base-measure (hyper-parameter) of the semantic tag indicator variable, which is not to be confused with a probability measure<sup>2</sup>

We update the indicator parameter via mean criteria,  $\psi_{v=w_i} = \sum_{i,j=1}^K \eta_{v=w_i}^{s_i | s_j} / (K^2)$ . If no prior on

<sup>2</sup>The base-measure used in Eq.(1) does not relate to a back-off model in LM sense. Here, instead of using a constant value for the hyper-parameters, we use probability scores that we obtain from LM.

### Algorithm 1 Markov Topic Regression

- 1: **for** each semantic tag topic  $s_k, k \leftarrow 1, \dots, K$  **do**
- 2:   – draw a topic mixture  $\phi_k \sim \text{Dir}(\beta_k | \lambda_k, \mathbf{x})$ ,
- 3:   – let  $\beta_k = \exp(f(\mathbf{x}; \lambda_k))$ ;  $\mathbf{x} = \{x_v\}_{v=1}^{V_i}$ ,  $\beta_k \in \mathcal{R}^{V_i}$
- 4: **for** each word  $w_v$  in vocabulary  $v \leftarrow 1, \dots, V$  **do**
- 5:   – draw a tag indicator mixture  $\psi_v \sim \text{Beta}(\eta)$ ,
- 6: **for** each utterance  $j \leftarrow 1, \dots, |U|$  **do**
- 7:   – draw transition distribution  $\theta_j^s \sim \text{Dir}(\alpha)$
- 8:   over states  $s_i$  and set  $c_{j1}=1$ .
- 9:   – **for** words  $w_i$  in  $u_j, i \leftarrow 1, \dots, N_j$  **do**
- 10:     – if  $i > 1$ , toss a coin  $c_{j,i} \sim \text{Binomial}(\psi_{w_i})$ .
- 11:     – if  $c_{j,i}=1$ , draw  $s_i \sim \text{Multi}(\theta_j^{s_i, s_{i-1}})$ <sup>†</sup>
- 12:     – otherwise  $s_i = s_{i-1}$ .
- 13:     – Sample  $w_i \sim \text{Multi}(\phi_{s_i})$ .

<sup>†</sup> Markov assumption over utterance words is used (See Eq.(4)).

a specific word exists, a default value is used for base measure,  $\eta_v = 0.01$ .

(IV) **Topic-Word Distribution Priors ( $\beta_k$ ):** Different from (Mimno et al., 2008), which uses asymmetric hyper-parameters on document-topic distributions, in MTR, we learn the asymmetric hyper-parameters of the semantic tag-word distributions. We use blocked Gibbs sampling, in which the topic assignments  $s_k$  and hyper-parameters  $\{\beta_k\}_{k=1}^K$  are alternately sampled at each Gibbs sampling lag period  $g$  given all other variables. We impose the prior knowledge on naturally related words, such that if two words "funny" and "hilarious" indicate the same given "genre" class, then their latent tag distributions should also be similar. We enforce this on smoothing parameter  $\beta_{k,v}$ , e.g.,  $\beta_{k, \text{'funny'}} \sim \beta_{k, \text{'hilarious'}}$  for a given tag  $k$  as follows:

At each  $g$  lag period of the Gibbs sampling,  $K$  log-linear models with parameters,  $\lambda_k^{(g)} \in \mathcal{R}^M$ , is trained to predict  $\beta_{kv}^{(g)} \in \beta_k$ , for each  $w_v$  of a tag  $s_k$ :

$$\beta_k^{(g)} = \exp(f(\mathbf{x}^l; \lambda_k^{(g)})) \quad (2)$$

where the log-linear function  $f$  is:

$$n_{kv}^{(g)} = f(\mathbf{x}_v^l; \lambda_k^{(g)}) = \sum_m \lambda_{k,m}^{(g)} x_{v,m}^l \quad (3)$$

Here  $\mathbf{x} \in \mathcal{R}^{V \times M}$  is the input matrix  $\mathbf{x}$ , wherein rows  $x_v \in \mathcal{R}^M$  represents  $M$ -dimensional scalar vector of explanatory features on vocabulary words. We use the word-tag posterior probabilities obtained from a CRF sequence model trained on labeled utterances as features. The  $\mathbf{x} = \{\mathbf{x}^l, \mathbf{x}^u\}$  has labeled ( $l$ ) and unlabeled ( $u$ ) parts. The labeled part contains  $V_l$  size vocabulary of which we know the semantic tags,  $\mathbf{x}^l = \{(x_{1^l, s_1}^l), \dots, (x_{V_l^l, s_{V_l}^l})\}$ . At the start of the Gibbs sampling, we designate the

$K$  latent topics to the  $K$  semantic tags of our labeled data. Therefore, we assign labeled words to their designated topics. This way we use observed scalar counts of each labeled word  $v$  associated with its semantic tag  $k$ ,  $n_{kv}^{(g)}$ , as the output label of its input vector,  $x_v^l$ ; an indication of likelihood of words getting sampled from the corresponding semantic label  $s_k$ . Since the impact of the asymmetric prior is equivalent to adding pseudo-counts to the sufficient statistics of the semantic tag to which the word belongs, we predict the pseudo-counts  $\beta_{kv}^{(g)}$  using the scalar counts of the labeled data,  $n_{kv}^{(g)}$ , based on the log-linear model in Eq. (2). At  $g=0$ , we use  $\beta_{kv}^{(0)}=2^8$ , if  $x_v \in X^l$ ; otherwise  $\beta_{kv}^{(0)}=2^{-2}$ , commonly used values for large and small  $\beta$ . Note that larger  $\beta$ -values indicate correlation between the word and the topic.

### 3.2 Collapsed Sampler

The goal of MTR is to infer the degree of relationship between a word  $v$  and each semantic tag  $k$ ,  $\phi_{kv}$ . To perform inference we need two components:

- a sampler which can draw from conditional  $P_{\text{MTR}}(s_{ji}=k|s_{ji-1}, s_{\setminus ji}, \alpha, \psi_i, \beta_{ji})$ , when  $c_{j,i}=1$ , where  $s_{ji}$  and  $s_{ji-1}$  are the semantic tags of the current  $w_i=v$  of vocabulary  $v$  and previous word  $w_{i-1}$  in utterance  $u_j$ , and  $s_{\setminus ji}$  are the semantic tag topics of all words except for  $w_i$ ; and,
- an estimation procedure for  $(\beta_{kv}, \lambda_k)$  (see §3.1).

We integrate out the multinomial and binomial parameters of the model: utterance-tag distributions  $\theta_j$ , binomial state transition indicator distribution per each word  $\psi_v$ , and  $\phi_k$  for tag-word distributions. We use collapsed Gibbs sampling to reduce random components and model the posterior distribution by obtaining samples  $(s_{ji}, c_{j,i})$  drawn from this distribution. Under the Markov assumption, for each word  $w_i=v$  in a given utterance  $u_j$ , if  $c_{j,i}=1$ , we sample a new tag  $s_i=k$  given the remaining tags and hyper-parameters  $\beta_k, \alpha$ , and  $\eta_{w_i=v}^{s_i|s_{i-1}}$ . Using the following parameters;  $n_{ji}^{(s_i)}$ , which is the number of words assigned to a semantic class  $s_i=k$  excluding case  $i$ , and  $n_{s_i}^{(s_{i-1})}$  is the number of transitions from class  $s_{i-1}$  to  $s_i$ , where indicator  $\mathbb{I}(s_{i-1}, s_i)=1$  if slot  $s_i=s_{i-1}$ , the update

equation is formulated as follows:

$$p(s_{ji} = k | \mathbf{w}, s_{-ji}, \alpha, \eta_{w_i}^{s_i|s_{i-1}}, \beta_k) \propto \frac{n_{ji}^{(s_i)} + \beta_{kw_i}}{n_{(\cdot)}^{(k)} + \sum_v \beta_{kv}} * (n_{s_i}^{(s_{i-1})} + \alpha) * \frac{(n_{s_{i+1}}^{(s_i)} + \mathbb{I}(s_{i-1}, s_i) + \mathbb{I}(s_{i+1}, s_i) + \alpha)}{n_{(\cdot)}^{(s_i)} + \mathbb{I}(s_{i-1}, k) + K\alpha} \quad (4)$$

## 4 Semi-Supervised Semantic Labeling

### 4.1 Semi Supervised Learning (SSL) with CRF

In (Subramanya et al., 2010), a new SSL method is described for adapting syntactic POS tagging of sentences in newswire articles along with search queries to a target domain of natural language (NL) questions. They decode unlabeled queries from target domain ( $t$ ) using a CRF model trained on the POS-labeled newswire data (source domain ( $o$ )). The unlabeled POS tag posteriors are then smoothed using a graph-based learning algorithm. On graph, the similarities are defined over sequences by constructing the graph over *types*, word 3-grams, where *types* capture the local context of words. Since CRF tagger only uses local features of the input to score tag pairs, they try to capture all the context with the graph with additional context features on *types*. Later, using viterbi decoding, they select the 1-best POS tag sequence,  $\mathbf{s}_j^*$  for each utterance  $u_j$ . Graph-based SSL defines a new CRF objective function:

$$\Lambda_{n+1}^{(t)} = \underset{\Lambda \in \mathcal{R}^K}{\text{argmin}} \left\{ - \sum_{j=1:l} \log p(\mathbf{s}_j | u_j; \Lambda_n^{(t)}) + \mu \|\Lambda_n^{(t)}\|^2 \right\} - \left\{ \tau \sum_{j=l}^{l+u} \log p_n(\mathbf{s}_j^* | u_j; \Lambda_n^{(t)}) \right\} \quad (5)$$

The first bracket in Eq.(5) is the loss on the labeled data and  $\mathcal{L}_2$  regularization on parameters,  $\Lambda_n^{(t)}$ , from  $n$ th iteration, same as standard CRF. The last term is the loss on unlabeled data from target domain with a hyper-parameter  $\tau$ . They use a small value for  $\tau$  to enable the new model to be as close as possible to the initial model trained on source data.

### 4.2 Retrospective Semi-Supervised CRF

We describe a *Retrospective* SSL (R-SSL) training with CRF (Algorithm 2), using MTR as a

smoothing model, instead of a graph-based model, as follows:

**I. DECODING and SMOOTHING.** The posterior probability of a tag  $s_{ji}=k$  given a word  $w_{ji}$  in unlabeled utterance  $u_j$  from target domain ( $t$ )  $\hat{p}_n(j, i)=\hat{p}_n(s_{ji}=k|w_{ji}; \Lambda_n^{(t)})$ , is decoded using the  $n$ -th iteration CRF model. MTR uses the decoded probabilities as semantic tag prior features on vocabulary items. We generate a word-tag matrix of posteriors,  $\mathbf{x} \in (0, 1)^{V \times K}$ , where  $K$  is the number of semantic tags and  $V$  is the vocabulary size from  $n$ -th iteration. Each row is a  $K$  dimensional vector of tag posterior probabilities  $x_v=\{x_{v1}, \dots, x_{vK}\}$  on the vocabulary term,  $w_v$ . The labeled rows  $\mathbf{x}^l$  of the vocabulary matrix,  $\mathbf{x}=\{\mathbf{x}^l, \mathbf{x}^u\}$ , contain only  $\{0, 1\}$  values, indicating the word’s observed semantic tags in the labeled data. Since a labeled term  $w_v$  can have different tags (e.g., ”clint eastwood” may be tagged as *actor-name* and *director-name* in the training data),  $\sum_k x_{vk} \geq 1$  holds. The  $\mathbf{x}$  is used as the input matrix of the  $k$ th log-linear model (corresponding to  $k$ th semantic tag (topic)) to infer the  $\beta$  hyper-parameter of MTR in Eq. (2). MTR generates smoothed conditional probabilities  $\phi_{kv}$  for each vocabulary term  $v$  given semantic tag  $k$ .

**II. INTERPOLATION.** For each word  $w_{ji}=v$  in unlabeled utterance  $u_j$ , we interpolate tag marginals from CRF and MTR for each semantic tag  $s_{ji} = k$ :

$$\hat{q}_n(s_{ji}|w_{ij}; \Lambda_n^{(t)}) = \pi \overbrace{\hat{p}_n(s_{ji}|w_{ij}; \Lambda_n^{(t)})}^{\text{CRF posterior}} + (1 - \pi) \overbrace{\phi_{kv}}^{\text{MTR}} \quad (6)$$

**III. VITERBI.** Using viterbi decoding over the tag marginals,  $\hat{q}_n(s_{ji}|w_{ij}; \Lambda_n^{(t)})$ , and transition probabilities obtained from the CRF model of  $n$ -th iteration, we get  $\hat{p}_n(s_j^*|u_j; \Lambda_n^{(t)})$ , the 1-best decode  $s_j^*$  of each unlabeled utterance  $u_j \in \mathcal{U}_n^u$ .

**IV. RETROSPECTIVE SSL (R-SSL).** After we decode the unlabeled data, we re-train a new CRF model at each iteration. Each iteration makes predictions on the semantic tags of unlabeled data with varying posterior probabilities. Motivated by (Lavoie et al., 2011), we want the loss function to have a dependency on the prior model predictions. Thus, R-SSL encodes the history of the prior pre-

---

## Algorithm 2 Retrospective Semi-Supervised CRF

---

**Input:** Labeled  $\mathcal{U}^l$ , and unlabeled  $\mathcal{U}^u$  data.

**Process:**  $\Lambda_n^{(o)} = \text{crf-train}(\mathcal{U}_l)$  at  $n=0, n=n+1$  †.

**While** not converged

$\hat{p} = \text{posterior-decode}(\mathcal{U}_n^u, \Lambda_n^{(o)})$

$\phi = \text{smooth-posteriors}(\hat{p})$  using MTR,

$\hat{q} = \text{interpolate-posteriors}(\hat{p}, \phi)$ ,

$\mathcal{U}_n^u = \text{viterbi-decode}(\hat{q})$

$\Lambda_{n+1}^{(t)} = \text{crf-retrospective}(\mathcal{U}^l, \mathcal{U}_n^u, \dots, \mathcal{U}_1^u, \Lambda_n^{(t)})$

† (n):iteration, (t):target, (o):source domains.

---

dictions, as follows:

$$\Lambda_{n+1}^{(t)} = \underset{\Lambda \in \mathcal{R}^K}{\text{argmin}} \left\{ \begin{aligned} & - \sum_{j=1:l} \log p(s_j|u_j; \Lambda_n^{(t)}) + \mu \|\Lambda_n^{(t)}\|^2 \\ & - \sum_{j=1:(l+u)} \max\{0, \hat{p}_n^{**}\} \end{aligned} \right\} \quad (7)$$

where,  $\hat{p}_n^{**} = 1 - \log h_n(u_j) \hat{p}_n(s_j^*|u_j; \Lambda_n^{(t)})$ . The first two terms are same as standard CRF. The last term ensures that the predictions of the current model have the same sign as the predictions of the previous models (using labeled and unlabeled data), denoted by a maximum margin hinge weight,  $h_n(u_j) = \frac{1}{n-1} \sum_{i=1}^{n-1} \hat{p}_n(s_j^*|u_j; \Lambda_n^{(t)})$ . It should also be noted that with MTR, the R-SSL learns the word-tag relations by using features that describe the words in context, eliminating the need for additional *type* representation of graph-based model. MTR provides a separate probability distribution  $\theta_j$  over tags for each utterance  $j$ , implicitly allowing for the same word  $v$  in separate utterances to differ in tag posteriors  $\phi_{kv}$ .

## 5 Experiments

### 5.1 Datasets and Tagsets

#### 5.1.1 Semantic Tagging Datasets

We focus here on audiovisual media in the *movie* domain. The user is expected to interact by voice with a system than can perform a variety of tasks such as browsing, searching, querying information, etc. To build initial NLU models for such a dialog system, we used crowd-sourcing to collect and annotate utterances, which we consider our source domain. Given movie domain-specific tasks, we asked the crowd about how they would

interact with the media system as if they were talking to a person.

Our data from target domain is internally collected from real-use scenarios of our spoken dialog system. The transcribed text forms of these utterances are obtained from speech recognition engine. Although the crowd-sourced data is similar to target domain, in terms of pre-defined user intentions, the target domain contains more descriptive vocabulary, which is almost twice as large as the source domain. This causes data-mismatch issues and hence provides a perfect test-bed for a domain adaptation task. In total, our corpus has a 40K semantically tagged utterances from each source and target domains. There are around 15 named-entity and 10 descriptive tags. We separated 5K utterances to test the performance of the semantic tagging models. The most frequent entities are: *movie-director* ('James Cameron'), *movie-title* ('Die Hard'), etc.; whereas top descriptive tags are: *genre* ('feel good'), *description* ('black and white', 'pg 13'), *review-rate* ('epic', 'not for me'), *theater-location* ('near me', 'city center'), etc.

Unlabeled utterances similar to the movie domain are pulled from a month old web query logs and extracted over 2 million search queries from well-known sites, e.g., IMDB, Netflix, etc. We filtered queries that are similar to our target set that start with *wh*-phrases ('what', 'who', etc.) as well as imperatives 'show', 'list', etc. In addition, we extracted web n-grams and entity lists (see §3) from movie related web sites, and online blogs and reviews. We collected around 300K movie review and blog entries on the entities observed in our data. We extract prior distributions for entities and n-grams to calculate entity list  $\eta$  and word-tag  $\beta$  priors (see §3.1).

### 5.1.2 Syntactic Tagging Datasets

We use the Wall Street Journal (WSJ) section of the Penn Treebank as our labeled source data. Following previous research, we train on sections 00-18, comprised of 38,219 POS-tagged sentences. To evaluate the domain adaptation (DA) approach and to compare with results reported by (Subramanya et al., 2010), we use the first and second half of QuestionBank (Judge et al., 2006) as our development and test sets (target). The QuestionBank contains 4000 POS-tagged questions, however it is difficult to tag with WSJ-trained taggers because the word order is different than WSJ

and contains a test-set vocabulary that is twice as large as the one in the development set. As for unlabeled data we crawled the web and collected around 100,000 questions that are similar in style and length to the ones in QuestionBank, e.g. "wh" questions. There are 36 different tag sets in the Penn dataset which includes tag labels for verbs, nouns, adjectives, adverbs, modal, determiners, prepositions, etc. More information about the Penn Tree-bank tag set can be found here (Marcus et al., 1993).

## 5.2 Models

We evaluated several baseline models on two tasks:

### 5.2.1 Semantic Clustering

Since **MTR** provides a mixture of properties adapted from earlier models, we present performance benchmarks on tag clustering using: (i) **LDA**; (ii) Hidden Markov Topic Model **HMTM** (Gruber et al., 2005); and, (iii) **w-LDA** (Petterson et al., 2010) that uses word features as priors in LDA. When a uniform  $\beta$  hyper-parameter is used with no external information on the state transitions in MTR, it reduces to a HMTM model. Similarly, if no Markov properties are used (bag-of-words), MTR reduces to w-LDA. Each topic model uses Gibbs sampling for inference and parameter learning. We sample models for 1000 iterations, with a 500-iteration burn-in and a sampling lag of 10. For testing we iterated the Gibbs sampler using the trained model for 10 iterations on the testing data.

### 5.2.2 SSL for Semantic/Syntactic Tagging

We evaluated three different baselines against our SSL models:

- \* **CRF**: a standard supervised sequence tagging.
- \* **Self-CRF**: a wrapper method for SSL using self-training. First a supervised learning algorithm is used to build a CRF model based on the labeled data. A CRF model is used to decode the unlabeled data to generate more labeled examples for re-training.
- \* **SSL-Graph**: A SSL model presented in (Subramanya et al., 2010) that uses graph-based learning as posterior tag smoother for CRF model using Eq.(5).

In addition to the three baseline, we evaluated three variations of our SSL method:

- \* **SSL-MTR**: Our first version of SSL uses MTR to

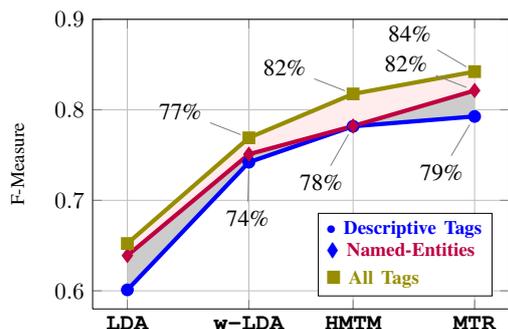


Figure 2: F-measure for semantic clustering performance. Performance differences for three different baseline models and our MTR approach by different semantic tags.

smooth the semantic tag posteriors of a unlabeled data decoded by the CRF model using Eq.(5).

★ **R-SSL-Graph:** Our second version uses graph-learning to smooth the tag posteriors and re-train a new CRF model using *retrospective* SSL in Eq.(7).

★ **R-SSL-MTR:** Our full model uses MTR as a Bayesian smoothing model, and retrospective SSL in Eq.(7) for iterative CRF training.

For all the CRF models, we use lexical features consisting of unigrams in a five-word window around the current word. To include contextual information, we add binary features for all possible tags. We inject dictionary constraints to all CRF models, such as features indicating label prior information. For each model we use several named entity features, e.g., *movie-title*, *actor-name*, etc., non-named entity (descriptive) features, e.g., *movie-description*, *movie-genre*, and domain independent dictionaries, e.g. *time*, *location*, etc. For graph-based learning, we implemented the algorithm presented in (Subramanya et al., 2010) and used the same hyper-parameters and features. For the rest of the hyper-parameters, we used:  $\alpha=0.01$  for MTR,  $\pi=0.5$  for interpolation mixing. These parameters were chosen based on the performance of the development set. All CRF objective functions were optimized using Stochastic Gradient Descent.

### 5.3 Results and Discussions

#### 5.3.1 Experiment 1: Clustering Semantic Tags.

Here, we want to demonstrate the performance of MTR model for capturing relationships between words and semantic tags against baseline topic

models: LDA, HMTM, w-LDA. We take the semantically labeled utterances from the movie target domain and use the first half for training and the rest for performance testing. We use all the collected unlabeled web queries from the movie domain. For fair comparison, each benchmark topic model is provided with prior information on word-semantic tag distributions based on the labeled training data, hence, each  $K$  latent topic is assigned to one of  $K$  semantic tags at the beginning of Gibbs sampling.

We evaluate the performance separately on descriptive tags, named-entities, and all tags together. The performance of the four topic models are reported in Figure 2. LDA shows the worst performance, even though some supervision is provided by way of labeled semantic tags. Although w-LDA improves semantic clustering performance over LDA, the fact that it does not have Markov properties makes it fall short behind MTR. As for the effect of word features in MTR, we see a 3% absolute performance gain over the second best performing HMTM baseline on named-entity tags, a 1% absolute gain on descriptive tags and a 2% absolute overall gain. As expected, we see a drop in F-measure on all models on descriptive tags.

#### 5.3.2 Experiment 2: Domain Adaptation Task.

We compare the performance of our SSL model to that of state-of-the-art models on semantic and syntactic tagging. Each SSL model is built using labeled training data from the source domain and unlabeled training data from target domain. In Table 2 we show the results on Movie and QuestionBank target test datasets. The results of SSL-Graph on QuestionBank is taken from (Subramanya et al., 2010). The self-training model, Self-CRF adds 3% improvement over supervised CRF models on movie domain, but does not improve syntactic tagging. Because it is always inherently biased towards the source domain, self-training tends to reinforce the knowledge that the supervised model already has. SSL-Graph works much better for both syntactic and semantic tagging compared to CRF and Self-CRF models. Our Bayesian MTR efficiently extracts information from the unlabeled data for the target domain. Combined with retrospective training, R-SSL-MTR demonstrates noticeable improvements,  $\sim 2\%$  on descriptive tags, and 1% absolute gains in overall semantic tag-

ging performance over SSL-Graph. On syntactic tagging, the two retrospective learning models is comparable, close to 1% improvement over the SSL-Graph and SSL-MTR.

Model	Movie Domain			QBank
	Desc.	NE	All	POS
<b>CRF</b>	75.05	75.84	75.84	83.80
<b>Self-CRF</b>	78.96	79.53	79.19	84.00
<b>SSL-Graph</b>	80.27	81.35	81.23	86.80
<b>SSL-MTR</b>	79.87	79.31	79.19	86.30
<b>R-SSL-Graph</b>	80.58	81.95	81.52	87.12
<b>R-SSL-MTR</b>	82.76	82.27	<b>82.24</b>	<b>87.34</b>

Table 2: Domain Adaptation performance in **F-measure** on Semantic Tagging on **Movie** Target domain and POS tagging on **QBank:QuestionBank**. Best performing models are **bolded**.

### 5.3.3 Experiment 3: Analysis of Semantic Disambiguation.

Here we focus on the accuracy of our models in tagging semantically ambiguous words. We investigate words that have more than one observed semantic tag in training data, such as "are there any [*war*]<sub>genre</sub> movies available.", "remove all movies about [*war*]<sub>description</sub>". Our corpus contained 30,000 unique vocabulary, 55% of which are contained in one or more semantic categories. Only 6.5% of those are tagged as multiple categories (polysemous), which are the sources of semantic ambiguity. Table-3 shows the precision of two best models for most confused words.

We compare our two best SSL models with different smoothing regularizes: R-SSL-MTR (**MTR**) and R-SSL-Graph (**GRAPH**). We use precision and recall criterion on semantically confused words.

In Table 3 we show two most frequent descriptive tags; *genre* and *description*, and commonly misclassified words by the two models. Results indicate that the R-SSL-MTR, performs better than the R-SSL-Graph, in activating the correct meaning of a word. The results indicate that incorporating context information with MTR is an effective option for identifying semantic ambiguity.

## 6 Conclusions

We have presented a novel semi supervised learning approach using a probabilistic clustering

Vocab.	genre		description	
	GRAPH	MTR	GRAPH	MTR
<i>war</i>	50%	100%	75%	88%
<i>popular</i>	90%	89%	80%	100%
<i>kids</i>	78%	86%	—	100%
<i>crime</i>	49%	80%	86%	67%
<i>zombie</i>	67%	89%	67%	86%

Table 3: Classification performance in F-measure for semantically ambiguous words on the most frequently confused descriptive tags in the movie domain.

method to semantically tag spoken language utterances. Our results show that encoding priors on words and context information contributes significantly to the performance of semantic clustering. We have also described an efficient iterative learning model that can handle data inconsistencies that leads to performance increases in semantic and syntactic tagging.

As a future work, we will investigate using session data, namely the entire dialog between the human and the computer. Rather than using single turn utterances, we hope to utilize the context information, e.g., information from previous turns for improving the performance of the semantic tagging of the current turns.

## References

- D. Blei, A. Ng, and M. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.
- J. Boyd-Graber, D. Blei, and X. Zhu. 2007. A topic model for word sense disambiguation. *Proc. EMNLP*.
- P.F. Brown, V.J.D. Pietra, P.V. deSouza, and J.C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.
- O. Chapelle, B. Scholkopf, and Alexander Zien. 2006. Semi-supervised learning. *MIT Press*.
- H. Daumé-III. 2010. Frustratingly easy semi-supervised domain adaptation. *Proc. Workshop on Domain Adaptation for Natural Language Processing at ACL*.
- T.L Griffiths, M. Steyvers, D.M. Blei, and J.M. Tenenbaum. 2005. Integrating topics and syntax. *Proc. of NIPS*.
- A. Gruber, M. Rosen-Zvi, and Y. Weiss. 2005. Hidden topic markov models. *Proc. of ICML*.
- H. Guo, H. Zhu, Z. Guo, X. Zhang, X. Wu, and Z. Su. 2009. Domain adaptation with latent semantic association for named entity recognition. *Proc. NAACL*.

- J. Judge, A. Cahill, and J. Van Genabith. 2006. Question-bank: Creating corpus of parse-annotated questions. *Proc. Int. Conf. Computational Linguistics and ACL*.
- A. Lavoie, M.E. Otey, N. Ratliff, and D. Sculley. 2011. History dependent domain adaptation. *Proc. NIPS Workshop on Domain Adaptation*.
- X. Li, Y.-Y. Wang, and A. Acero. 2009. Extracting structured information from user queries with semi-supervised conditional random fields. *Proc. of SIGIR*.
- L. Li, B. Roth, and C. Sporleder. 2010. Topic models for word sense disambiguation and token-based idiom detection. *Proc. ACL*.
- X. Li. 2010. Understanding semantic structure of noun phrase queries. *Proc. ACL*.
- J. Liu, X. Li, A. Acero, and Ye-Yi Wang. 2011. Lexicon modeling for query understanding. *Proc. of ICASSP*.
- M. P. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 27:1–30.
- D. Mimno, W. Li, and A. McCallum. 2008. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. *Proc. UAI*.
- T. Moon, K. Erk, and J. Baldridge. 2010. Crouching dirichlet, hidden markov model: Unsupervised pos tagging with context local tag generation. *Proc. ACL*.
- V.-A. Nyugen, J. Boyd-Graber, and P. Resnik. 2012. Sits: A hierarchical nonparametric model using speaker identity for topic segmentation in multiparty conversations. *Proc. ACL*.
- P. Pantel. 2003. Clustering by committee. *Ph.D. Thesis, University of Alberta, Edmonton, Alta., Canada*.
- J. Petterson, A. Smola, T. Caetano, W. Buntine, and S. Narayanamurthy. 2010. Word features for latent dirichlet allocation. In *Proc. NIPS*.
- J. Reisinger and R. Mooney. 2011. Cross-cutting models of lexical semantics. In *Proc. of EMNLP*.
- S. Singh, D. Hillard, and C. Leggetter. 2010. Minimally-supervised extraction of entities from text advertisements. *Proc. NAACL-HLT*.
- A. Stolcke. 2002. An extensible language modeling toolkit. *Proc. Interspeech*.
- A. Subramanya, S. Petrov, and F. Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proc. EMNLP*.
- G. Tur and R. DeMori. 2011. Spoken language understanding: Systems for extracting semantic information from speech. *Wiley Press*.
- Y.-Y. Wang, R. Hoffman, X. Li, and J. Szymanski. 2009. Semi-supervised learning of semantic classes for query understanding from the web and for the web. In *The 18th ACM Conference on Information and Knowledge Management*.
- X. Zhu. 2005. Semi-supervised learning literature survey. *Technical Report 1530, University of Wisconsin-Madison*.

# Parsing Graphs with Hyperedge Replacement Grammars

**David Chiang**

Information Sciences Institute  
University of Southern California

**Jacob Andreas**

Columbia University  
University of Cambridge

**Daniel Bauer**

Department of Computer Science  
Columbia University

**Karl Moritz Hermann**

Department of Computer Science  
University of Oxford

**Bevan Jones**

University of Edinburgh  
Macquarie University

**Kevin Knight**

Information Sciences Institute  
University of Southern California

## Abstract

Hyperedge replacement grammar (HRG) is a formalism for generating and transforming graphs that has potential applications in natural language understanding and generation. A recognition algorithm due to Lautemann is known to be polynomial-time for graphs that are connected and of bounded degree. We present a more precise characterization of the algorithm's complexity, an optimization analogous to binarization of context-free grammars, and some important implementation details, resulting in an algorithm that is practical for natural-language applications. The algorithm is part of *Bolinas*, a new software toolkit for HRG processing.

## 1 Introduction

Hyperedge replacement grammar (HRG) is a context-free rewriting formalism for generating graphs (Drewes et al., 1997), and its synchronous counterpart can be used for transforming graphs to/from other graphs or trees. As such, it has great potential for applications in natural language understanding and generation, and semantics-based machine translation (Jones et al., 2012). Figure 1 shows some examples of graphs for natural-language semantics.

A polynomial-time recognition algorithm for HRGs was described by Lautemann (1990), building on the work of Rozenberg and Welzl (1986) on boundary node label controlled grammars, and others have presented polynomial-time algorithms as well (Mazanek and Minas, 2008; Moot, 2008). Although Lautemann's algorithm is correct and

tractable, its presentation is prefaced with the remark: "As we are only interested in distinguishing polynomial time from non-polynomial time, the analysis will be rather crude, and implementation details will be explicated as little as possible." Indeed, the key step of the algorithm, which matches a rule against the input graph, is described at a very high level, so that it is not obvious (for a non-expert in graph algorithms) how to implement it. More importantly, this step as described leads to a time complexity that is polynomial, but potentially of very high degree.

In this paper, we describe in detail a more efficient version of this algorithm and its implementation. We give a more precise complexity analysis in terms of the grammar and the size and maximum degree of the input graph, and we show how to optimize it by a process analogous to binarization of CFGs, following Gildea (2011). The resulting algorithm is practical and is implemented as part of the open-source Bolinas toolkit for hyperedge replacement grammars.

## 2 Hyperedge replacement grammars

We give a short example of how HRG works, followed by formal definitions.

### 2.1 Example

Consider a weighted graph language involving just two types of semantic frames (*want* and *believe*), two types of entities (*boy* and *girl*), and two roles (*arg0* and *arg1*). Figure 1 shows a few graphs from this language.

Figure 2 shows how to derive one of these graphs using an HRG. The derivation starts with a single edge labeled with the nonterminal symbol  $S$ . The first rewriting step replaces this edge with a subgraph, which we might read as "The

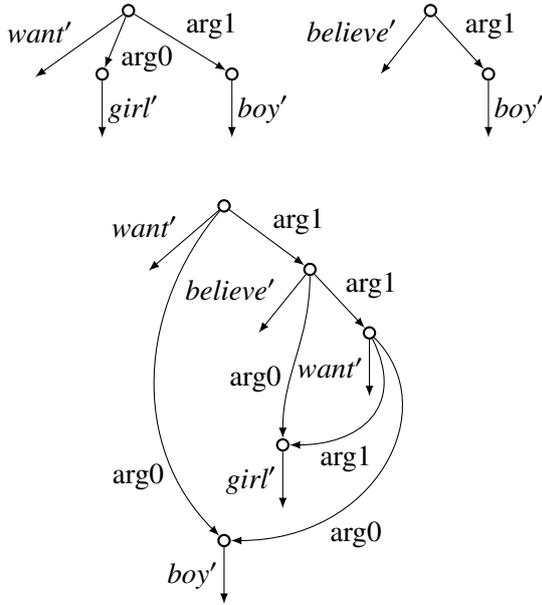


Figure 1: Sample members of a graph language, representing the meanings of (clockwise from upper left): “The girl wants the boy,” “The boy is believed,” and “The boy wants the girl to believe that he wants her.”

boy wants something ( $X$ ) involving himself.” The second rewriting step replaces the  $X$  edge with another subgraph, which we might read as “The boy wants the girl to believe something ( $Y$ ) involving both of them.” The derivation continues with a third rewriting step, after which there are no more nonterminal-labeled edges.

## 2.2 Definitions

The graphs we use in this paper have edge labels, but no node labels; while node labels are intuitive for many graphs in NLP, using both node and edge labels complicates the definition of hyperedge grammar and algorithms. All of our graphs are directed (ordered), as the purpose of most graph structures in NLP is to model dependencies between entities.

**Definition 1.** An *edge-labeled, ordered hypergraph* is a tuple  $H = \langle V, E, \ell \rangle$ , where

- $V$  is a finite set of nodes
- $E \subseteq V^+$  is a finite set of hyperedges, each of which connects one or more distinct nodes
- $\ell : E \rightarrow C$  assigns a label (drawn from the finite set  $C$ ) to each edge.

For brevity we use the terms *graph* and *hypergraph* interchangeably, and similarly for *edge* and *hyperedge*. In the definition of HRGs, we will use the notion of hypergraph fragments, which are the elementary structures that the grammar assembles into hypergraphs.

**Definition 2.** A *hypergraph fragment* is a tuple  $\langle V, E, \ell, X \rangle$ , where  $\langle V, E, \ell \rangle$  is a hypergraph and  $X \in V^+$  is a list of distinct nodes called the *external nodes*.

The function of graph fragments in HRG is analogous to the right-hand sides of CFG rules and to elementary trees in tree adjoining grammars (Joshi and Schabes, 1997). The external nodes indicate how to integrate a graph into another graph during a derivation, and are analogous to foot nodes. In diagrams, we draw them with a black circle ( $\bullet$ ).

**Definition 3.** A *hyperedge replacement grammar* (HRG) is a tuple  $G = \langle N, T, P, S \rangle$  where

- $N$  and  $T$  are finite disjoint sets of nonterminal and terminal symbols
- $S \in N$  is the start symbol
- $P$  is a finite set of productions of the form  $A \rightarrow R$ , where  $A \in N$  and  $R$  is a graph fragment over  $N \cup T$ .

We now describe the HRG rewriting mechanism.

**Definition 4.** Given a HRG  $G$ , we define the relation  $H \Rightarrow_G H'$  (or,  $H'$  is derived from  $H$  in one step) as follows. Let  $e = (v_1 \cdots v_k)$  be an edge in  $H$  with label  $A$ . Let  $(A \rightarrow R)$  be a production of  $G$ , where  $R$  has external nodes  $X_R = (u_1 \cdots u_k)$ . Then we write  $H \Rightarrow_G H'$  if  $H'$  is the graph formed by removing  $e$  from  $H$ , making an isomorphic copy of  $R$ , and identifying  $v_i$  with (the copy of)  $u_i$  for  $i = 1, \dots, k$ .

Let  $H \Rightarrow_G^* H'$  (or,  $H'$  is derived from  $H$ ) be the reflexive, transitive closure of  $\Rightarrow_G$ . The *graph language* of a grammar  $G$  is the (possibly infinite) set of graphs  $H$  that have no edges with nonterminal labels such that

$$\begin{array}{c} \circ \\ \downarrow S \\ \bullet \end{array} \Rightarrow_G^* H.$$

When a HRG rule  $(A \rightarrow R)$  is applied to an edge  $e$ , the mapping of external nodes in  $R$  to the

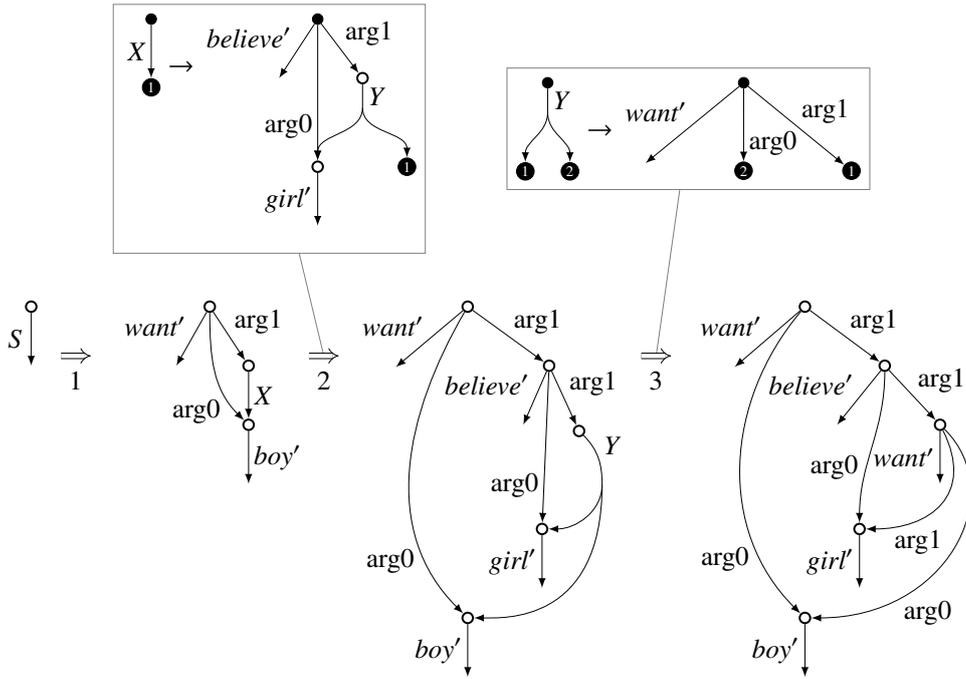


Figure 2: Derivation of a hyperedge replacement grammar for a graph representing the meaning of “The boy wants the girl to believe that he wants her.”

nodes of  $e$  is implied by the ordering of nodes in  $e$  and  $X_R$ . When writing grammar rules, we make this ordering explicit by writing the left hand side of a rule as an edge and indexing the external nodes of  $R$  on both sides, as shown in Figure 2.

HRG derivations are context-free in the sense that the applicability of each production depends on the nonterminal label of the replaced edge only. This allows us to represent a derivation as a derivation tree, and sets of derivations of a graph as a derivation forest (which can in turn be represented as hypergraphs). Thus we can apply many of the methods developed for other context free grammars. For example, it is easy to define weighted and synchronous versions of HRGs.

**Definition 5.** If  $K$  is a semiring, a  $K$ -weighted HRG is a tuple  $G = \langle N, T, P, S, \lambda \rangle$ , where  $\langle N, T, P, S \rangle$  is a HRG and  $\lambda : P \rightarrow K$  assigns a weight in  $K$  to each production. The weight of a derivation of  $G$  is the product of the weights of the productions used in the derivation.

We defer a definition of synchronous HRGs until Section 4, where they are discussed in detail.

### 3 Parsing

Lautemann’s recognition algorithm for HRGs is a generalization of the CKY algorithm for CFGs.

Its key step is the matching of a rule against the input graph, analogous to the concatenation of two spans in CKY. The original description leaves open how this matching is done, and because it tries to match the whole rule at once, it has asymptotic complexity exponential in the number of non-terminal edges. In this section, we present a refinement that makes the rule-matching procedure explicit, and because it matches rules little by little, similarly to binarization of CFG rules, it does so more efficiently than the original.

Let  $H$  be the input graph. Let  $n$  be the number of nodes in  $H$ , and  $d$  be the maximum degree of any node. Let  $G$  be a HRG. For simplicity, we assume that the right-hand sides of rules are connected. This restriction entails that each graph generated by  $G$  is connected; therefore, we assume that  $H$  is connected as well. Finally, let  $m$  be an arbitrary node of  $H$  called the *marker node*, whose usage will become clear below.<sup>1</sup>

#### 3.1 Representing subgraphs

Just as CKY deals with substrings  $(i, j]$  of the input, the HRG parsing algorithm deals with edge-induced subgraphs  $I$  of the input. An edge-induced subgraph of  $H = \langle V, E, \ell \rangle$  is, for some

<sup>1</sup>To handle the more general case where  $H$  is not connected, we would need a marker for each component.

subset  $E' \subseteq E$ , the smallest subgraph containing all edges in  $E'$ . From now on, we will assume that all subgraphs are edge-induced subgraphs.

In CKY, the two endpoints  $i$  and  $j$  completely specify the recognized part of the input,  $w_{i+1} \cdots w_j$ . Likewise, we do not need to store all of  $I$  explicitly.

**Definition 6.** Let  $I$  be a subgraph of  $H$ . A *boundary node* of  $I$  is a node in  $I$  which is either a node with an edge in  $H \setminus I$  or an external node. A *boundary edge* of  $I$  is an edge in  $I$  which has a boundary node as an endpoint. The *boundary representation* of  $I$  is the tuple  $\langle bn(I), be(I, v), m \in I \rangle$ , where

- $bn(I)$  is the set of boundary nodes of  $I$
- $be(I, v)$  be the set of boundary edges of  $v$  in  $I$
- $(m \in I)$  is a flag indicating whether the marker node is in  $I$ .

The boundary representation of  $I$  suffices to specify  $I$  compactly.

**Proposition 1.** *If  $I$  and  $I'$  are two subgraphs of  $H$  with the same boundary representation, then  $I = I'$ .*

*Proof.* Case 1:  $bn(I)$  is empty. If  $m \in I$  and  $m \in I'$ , then all edges of  $H$  must belong to both  $I$  and  $I'$ , that is,  $I = I' = H$ . Otherwise, if  $m \notin I$  and  $m \notin I'$ , then no edges can belong to either  $I$  or  $I'$ , that is,  $I = I' = \emptyset$ .

Case 2:  $bn(I)$  is nonempty. Suppose  $I \neq I'$ ; without loss of generality, suppose that there is an edge  $e$  that is in  $I \setminus I'$ . Let  $\pi$  be the shortest path (ignoring edge direction) that begins with  $e$  and ends with a boundary node. All the edges along  $\pi$  must be in  $I \setminus I'$ , or else there would be a boundary node in the middle of  $\pi$ , and  $\pi$  would not be the shortest path from  $e$  to a boundary node. Then, in particular, the last edge of  $\pi$  must be in  $I \setminus I'$ . Since it has a boundary node as an endpoint, it must be a boundary edge of  $I$ , but cannot be a boundary edge of  $I'$ , which is a contradiction.  $\square$

If two subgraphs are disjoint, we can use their boundary representations to compute the boundary representation of their union.

**Proposition 2.** *Let  $I$  and  $J$  be two subgraphs whose edges are disjoint. A node  $v$  is a boundary node of  $I \cup J$  iff one of the following holds:*

- (i)  $v$  is a boundary node of one subgraph but not the other

- (ii)  $v$  is a boundary node of both subgraphs, and has an edge which is not a boundary edge of either.

*An edge is a boundary edge of  $I \cup J$  iff it has a boundary node of  $I \cup J$  as an endpoint and is a boundary edge of  $I$  or  $J$ .*

*Proof.*  $(\Rightarrow)$   $v$  has an edge in either  $I$  or  $J$  and an edge  $e$  outside both  $I$  and  $J$ . Therefore it must be a boundary node of either  $I$  or  $J$ . Moreover,  $e$  is not a boundary edge of either, satisfying condition (ii).

$(\Leftarrow)$  Case (i): without loss of generality, assume  $v$  is a boundary node of  $I$ . It has an edge  $e$  in  $I$ , and therefore in  $I \cup J$ , and an edge  $e'$  outside  $I$ , which must also be outside  $J$ . For  $e' \notin J$  (because  $I$  and  $J$  are disjoint), and if  $e' \in J$ , then  $v$  would be a boundary node of  $J$ . Therefore,  $e' \notin I \cup J$ , so  $v$  is a boundary node of  $I \cup J$ . Case (ii):  $v$  has an edge in  $I$  and therefore  $I \cup J$ , and an edge not in either  $I$  or  $J$ .  $\square$

This result leads to Algorithm 1, which runs in time linear in the number of boundary nodes.

---

**Algorithm 1** Compute the union of two disjoint subgraphs  $I$  and  $J$ .

---

```

for all  $v \in bn(I)$  do
   $E \leftarrow be(I, v) \cup be(J, v)$ 
  if  $v \notin bn(J)$  or  $v$  has an edge not in  $E$  then
    add  $v$  to  $bn(I \cup J)$ 
     $be(I \cup J, v) \leftarrow E$ 
for all  $v \in bn(J)$  do
  if  $v \notin bn(I)$  then
    add  $v$  to  $bn(I \cup J)$ 
     $be(I \cup J, v) \leftarrow be(I, v) \cup be(J, v)$ 
 $(m \in I \cup J) \leftarrow (m \in I) \vee (m \in J)$ 

```

---

In practice, for small subgraphs, it may be more efficient simply to use an explicit set of edges instead of the boundary representation. For the Geo-Query corpus (Tang and Mooney, 2001), whose graphs are only 7.4 nodes on average, we generally find this to be the case.

### 3.2 Treewidth

Lautemann's algorithm tries to match a rule against the input graph all at once. But we can optimize the algorithm by matching a rule incrementally. This is analogous to the rank-minimization problem for linear context-free rewriting systems. Gildea has shown that this problem is related to

the notion of *treewidth* (Gildea, 2011), which we review briefly here.

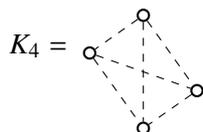
**Definition 7.** A *tree decomposition* of a graph  $H = \langle V, E \rangle$  is a tree  $T$ , each of whose nodes  $\eta$  is associated with sets  $V_\eta \subseteq V$  and  $E_\eta \subseteq E$ , with the following properties:

1. Vertex cover: For each  $v \in V$ , there is a node  $\eta \in T$  such that  $v \in V_\eta$ .
2. Edge cover: For each  $e = (v_1 \cdots v_k) \in E$ , there is exactly one node  $\eta \in T$  such that  $e \in E_\eta$ . We say that  $\eta$  *introduces*  $e$ . Moreover,  $v_1, \dots, v_k \in V_\eta$ .
3. Running intersection: For each  $v \in V$ , the set  $\{\eta \in T \mid v \in V_\eta\}$  is connected.

The *width* of  $T$  is  $\max |V_\eta| - 1$ . The *treewidth* of  $H$  is the minimal width of any tree decomposition of  $H$ .

A tree decomposition of a graph fragment  $\langle V, E, X \rangle$  is a tree decomposition of  $\langle V, E \rangle$  that has the additional property that all the external nodes belong to  $V_\eta$  for some  $\eta$ . (Without loss of generality, we assume that  $\eta$  is the root.)

For example, Figure 3b shows a graph, and Figure 3c shows a tree decomposition. This decomposition has width three, because its largest node has 4 elements. In general, a tree has width one, and it can be shown that a graph has treewidth at most two iff it does not have the following graph as a minor (Bodlaender, 1997):



Finding a tree decomposition with minimal width is in general NP-hard (Arnborg et al., 1987). However, we find that for the graphs we are interested in in NLP applications, even a naïve algorithm gives tree decompositions of low width in practice: simply perform a depth-first traversal of the edges of the graph, forming a tree  $T$ . Then, augment the  $V_\eta$  as necessary to satisfy the running intersection property.

As a test, we extracted rules from the GeoQuery corpus (Tang and Mooney, 2001) using the SYNSEM algorithm (Jones et al., 2012), and computed tree decompositions exactly using a branch-and-bound method (Gogate and Dechter, 2004) and this approximate method. Table 1 shows that, in practice, treewidths are not very high even when computed only approximately.

method	mean	max
exact	1.491	2
approximate	1.494	3

Table 1: Mean and maximum treewidths of rules extracted from the GeoQuery corpus, using exact and approximate methods.

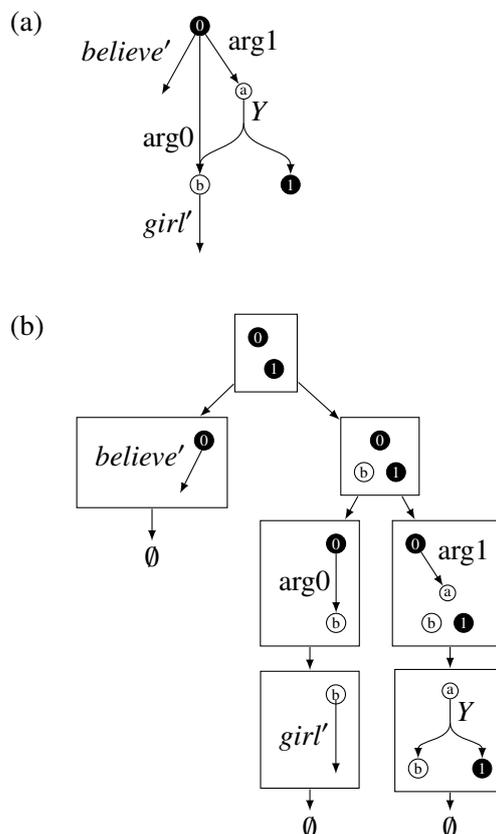


Figure 3: (a) A rule right-hand side, and (b) a nice tree decomposition.

Any tree decomposition can be converted into one which is *nice* in the following sense (simplified from Cygan et al. (2011)). Each tree node  $\eta$  must be one of:

- A leaf node, such that  $V_\eta = \emptyset$ .
- A unary node, which introduces exactly one edge  $e$ .
- A binary node, which introduces no edges.

The example decomposition in Figure 3c is nice. This canonical form simplifies the operation of the parser described in the following section.

Let  $G$  be a HRG. For each production  $(A \rightarrow R) \in G$ , find a nice tree decomposition of  $R$  and call it  $T_R$ . The treewidth of  $G$  is the maximum

treewidth of any right-hand side in  $G$ .

The basic idea of the recognition algorithm is to recognize the right-hand side of each rule incrementally by working bottom-up on its tree decomposition. The properties of tree decomposition allow us to limit the number of boundary nodes of the partially-recognized rule.

More formally, let  $R_{\succeq\eta}$  be the subgraph of  $R$  induced by the union of  $E_{\eta'}$  for all  $\eta'$  equal to or dominated by  $\eta$ . Then we can show the following.

**Proposition 3.** *Let  $R$  be a graph fragment, and assume a tree decomposition of  $R$ . All the boundary nodes of  $R_{\succeq\eta}$  belong to  $V_\eta \cap V_{parent(\eta)}$ .*

*Proof.* Let  $v$  be a boundary node of  $R_{\succeq\eta}$ . Node  $v$  must have an edge in  $R_{\succeq\eta}$  and therefore in  $R_{\eta'}$  for some  $\eta'$  dominated by or equal to  $\eta$ .

Case 1:  $v$  is an external node. Since the root node contains all the external nodes, by the running intersection property, both  $V_\eta$  and  $V_{parent(\eta)}$  must contain  $v$  as well.

Case 2:  $v$  has an edge not in  $R_{\succeq\eta}$ . Therefore there must be a tree node not dominated by or equal to  $\eta$  that contains this edge, and therefore  $v$ . So by the running intersection property,  $\eta$  and its parent must contain  $v$  as well.  $\square$

This result, in turn, will allow us to bound the complexity of the parsing algorithm in terms of the treewidth of  $G$ .

### 3.3 Inference rules

We present the parsing algorithm as a deductive system (Shieber et al., 1995). The items have one of two forms. A *passive* item has the form  $[A, I, X]$ , where  $X \in V^+$  is an explicit ordering of the boundary nodes of  $I$ . This means that we have recognized that  $A \Rightarrow_G^* I$ . Thus, the goal item is  $[S, H, \epsilon]$ . An *active* item has the form  $[A \rightarrow R, \eta, I, \phi]$ , where

- $(A \rightarrow R)$  is a production of  $G$
- $\eta$  is a node of  $T_R$
- $I$  is a subgraph of  $H$
- $\phi$  is a bijection between the boundary nodes of  $R_{\succeq\eta}$  and those of  $I$ .

The parser must ensure that  $\phi$  is a bijection when it creates a new item. Below, we use the notation  $\{e \mapsto e'\}$  or  $\{e \mapsto X\}$  for the mapping that sends each node of  $e$  to the corresponding node of  $e'$  or  $X$ .

Passive items are generated by the following rule:

- Root 
$$\frac{[B \rightarrow Q, \theta, J, \psi]}{[B, J, X]}$$

where  $\theta$  is the root of  $T_Q$ , and  $X_j = \psi(X_{Q,j})$ .

If we assume that the  $T_R$  are nice, then the inference rules that generate active items follow the different types of nodes in a nice tree decomposition:

- Leaf 
$$\overline{[A \rightarrow R, \eta, \emptyset, \emptyset]}$$

where  $\eta$  is a leaf node of  $T_R$ .

- (Unary) Nonterminal

$$\frac{[A \rightarrow R, \eta_1, I, \phi] \quad [B, J, X]}{[A \rightarrow R, \eta, I \cup J, \phi \cup \{e \mapsto X\}]}$$

where  $\eta_1$  is the only child of  $\eta$ , and  $e$  is introduced by  $\eta$  and is labeled with nonterminal  $B$ .

- (Unary) Terminal

$$\frac{[A \rightarrow R, \eta_1, I, \phi]}{[A \rightarrow R, \eta, I \cup \{e'\}, \phi \cup \{e \mapsto e'\}]}$$

where  $\eta_1$  is the only child of  $\eta$ ,  $e$  is introduced by  $\eta$ , and  $e$  and  $e'$  are both labeled with terminal  $a$ .

- Binary

$$\frac{[A \rightarrow R, \eta_1, I, \phi_1] \quad [A \rightarrow R, \eta_2, J, \phi_2]}{[A \rightarrow R, \eta, I \cup J, \phi_1 \cup \phi_2]}$$

where  $\eta_1$  and  $\eta_2$  are the two children of  $\eta$ .

In the Nonterminal, Terminal, and Binary rules, we form unions of subgraphs and unions of mappings. When forming the union of two subgraphs, we require that the subgraphs be disjoint (however, see Section 3.4 below for a relaxation of this condition). When forming the union of two mappings, we require that the result be a bijection. If either of these conditions is not met, the inference rule cannot apply.

For efficiency, it is important to index the items for fast access. For the Nonterminal inference rule, passive items  $[B, J, X]$  should be indexed by key  $\langle B, |bn(J)| \rangle$ , so that when the next item on the agenda is an active item  $[A \rightarrow R, \eta_1, I, \phi]$ , we know that all possible matching passive items are

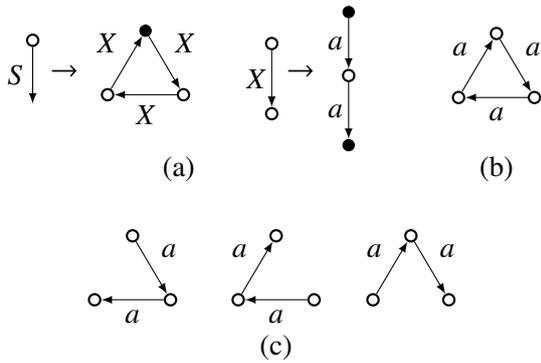


Figure 4: Illustration of unsoundness in the recognition algorithm without the disjointness check. Using grammar (a), the recognition algorithm would incorrectly accept the graph (b) by assembling together the three overlapping fragments (c).

under key  $\langle \ell(e), |e| \rangle$ . Similarly, active items should be indexed by key  $\langle \ell(e), |e| \rangle$  so that they can be found when the next item on the agenda is a passive item. For the Binary inference rule, active items should be indexed by their tree node ( $\eta_1$  or  $\eta_2$ ).

This procedure can easily be extended to produce a packed forest of all possible derivations of the input graph, representable as a hypergraph just as for other context-free rewriting formalisms. The Viterbi algorithm can then be applied to this representation to find the highest-probability derivation, or the Inside/Outside algorithm to set weights by Expectation-Maximization.

### 3.4 The disjointness check

A successful proof using the inference rules above builds an HRG derivation (comprising all the rewrites used by the Nonterminal rule) which derives a graph  $H'$ , as well as a graph isomorphism  $\phi : H' \rightarrow H$  (the union of the mappings from all the items).

During inference, whenever we form the union of two subgraphs, we require that the subgraphs be disjoint. This is a rather expensive operation: it can be done using only their boundary representations, but the best algorithm we are aware of is still quadratic in the number of boundary nodes.

Is it possible to drop the disjointness check? If we did so, it would become possible for the algorithm to recognize the same part of  $H$  twice. For example, Figure 4 shows an example of a grammar and an input that would be incorrectly recognized.

However, we can replace the disjointness check

with a weaker and faster check such that any derivation that merges two non-disjoint subgraphs will ultimately fail, and therefore the derived graph  $H'$  is isomorphic to the input graph  $H'$  as desired. This weaker check is to require, when merging two subgraphs  $I$  and  $J$ , that:

1.  $I$  and  $J$  have no boundary edges in common, and
2. If  $m$  belongs to both  $I$  and  $J$ , it must be a boundary node of both.

Condition (1) is enough to guarantee that  $\phi$  is *locally one-to-one* in the sense that for all  $v \in H'$ ,  $\phi$  restricted to  $v$  and its neighbors is one-to-one. This is easy to show by induction: if  $\phi_I : I' \rightarrow H$  and  $\phi_J : J' \rightarrow H$  are locally one-to-one, then  $\phi_I \cup \phi_J$  must also be, provided condition (1) is met. Intuitively, the consequence of this is that we can detect any place where  $\phi$  changes (say) from being one-to-one to two-to-one. So if  $\phi$  is two-to-one, then it must be two-to-one everywhere (as in the example of Figure 4).

But condition (2) guarantees that  $\phi$  maps only one node to the marker  $m$ . We can show this again by induction: if  $\phi_I$  and  $\phi_J$  each map only one node to  $m$ , then  $\phi_I \cup \phi_J$  must map only one node to  $m$ , by a combination of condition (2) and the fact that the inference rules guarantee that  $\phi_I$ ,  $\phi_J$ , and  $\phi_I \cup \phi_J$  are one-to-one on boundary nodes.

Then we can show that, since  $m$  is recognized exactly once, the whole graph is also recognized exactly once.

**Proposition 4.** *If  $H$  and  $H'$  are connected graphs,  $\phi : H' \rightarrow H$  is locally one-to-one, and  $\phi^{-1}$  is defined for some node of  $H$ , then  $\phi$  is a bijection.*

*Proof.* Suppose that  $\phi$  is not a bijection. Then there must be two nodes  $v'_1, v'_2 \in H'$  such that  $\phi(v'_1) = \phi(v'_2) = v \in H$ . We also know that there is a node, namely,  $m$ , such that  $m' = \phi^{-1}(m)$  is defined.<sup>2</sup> Choose a path  $\pi$  (ignoring edge direction) from  $v$  to  $m$ . Because  $\phi$  is a local isomorphism, we can construct a path from  $v'_1$  to  $m'$  that maps to  $\pi$ . Similarly, we can construct a path from  $v'_2$  to  $m'$  that maps to  $\pi$ . Let  $u'$  be the first node that these two paths have in common. But  $u'$  must have two edges that map to the same edge, which is a contradiction.  $\square$

<sup>2</sup>If  $H$  were not connected, we would choose the marker in the same connected component as  $v$ .

### 3.5 Complexity

The key to the efficiency of the algorithm is that the treewidth of  $G$  leads to a bound on the number of boundary nodes we must keep track of at any time.

Let  $k$  be the treewidth of  $G$ . The time complexity of the algorithm is the number of ways of instantiating the inference rules. Each inference rule mentions only boundary nodes of  $R_{\geq \eta}$  or  $R_{\geq \eta_i}$ , all of which belong to  $V_\eta$  (by Proposition 3), so there are at most  $|V_\eta| \leq k + 1$  of them. In the Nonterminal and Binary inference rules, each boundary edge could belong to  $I$  or  $J$  or neither. Therefore, the number of possible instantiations of any inference rule is in  $O((3^d n)^{k+1})$ .

The space complexity of the algorithm is the number of possible items. For each active item  $[A \rightarrow R, \eta, I, \phi]$ , every boundary node of  $R_{\geq \eta}$  must belong to  $V_\eta \cap V_{parent(\eta)}$  (by Proposition 3). Therefore the number of boundary nodes is at most  $k + 1$  (but typically less), and the number of possible items is in  $O((2^d n)^{k+1})$ .

## 4 Synchronous Parsing

As mentioned in Section 2.2, because HRGs have context-free derivation trees, it is easy to define *synchronous HRGs*, which define mappings between languages of graphs.

**Definition 8.** A *synchronous hyperedge replacement grammar* (SHRG) is a tuple  $G = \langle N, T, T', P, S \rangle$ , where

- $N$  is a finite set of nonterminal symbols
- $T$  and  $T'$  are finite sets of terminal symbols
- $S \in N$  is the start symbol
- $P$  is a finite set of productions of the form  $(A \rightarrow \langle R, R', \sim \rangle)$ , where  $R$  is a graph fragment over  $N \cup T$  and  $R'$  is a graph fragment over  $N \cup T'$ . The relation  $\sim$  is a bijection linking nonterminal mentions in  $R$  and  $R'$ , such that if  $e \sim e'$ , then they have the same label. We call  $R$  the *source* side and  $R'$  the *target* side.

Some NLP applications (for example, word alignment) require *synchronous parsing*: given a pair of graphs, finding the derivation or forest of derivations that simultaneously generate both the source and target. The algorithm to do this is a straightforward generalization of the HRG parsing

algorithm. For each rule  $(A \rightarrow \langle R, R', \sim \rangle)$ , we construct a nice tree decomposition of  $R \cup R'$  such that:

- All the external nodes of both  $R$  and  $R'$  belong to  $V_\eta$  for some  $\eta$ . (Without loss of generality, assume that  $\eta$  is the root.)
- If  $e \sim e'$ , then  $e$  and  $e'$  are introduced by the same tree node.

In the synchronous parsing algorithm, passive items have the form  $[A, I, X, I', X']$  and active items have the form  $[A \rightarrow R : R', \eta, I, \phi, I', \phi']$ . For brevity we omit a re-presentation of all the inference rules, as they are very similar to their non-synchronous counterparts. The main difference is that in the Nonterminal rule, two linked edges are rewritten simultaneously:

$$\frac{[A \rightarrow R : R', \eta_1, I, \phi, I', \phi'] \quad [B, J, X, J', X']}{[A \rightarrow R : R', \eta, I \cup J, \phi \cup \{e_j \mapsto X_j\}, I' \cup J', \phi' \cup \{e'_j \mapsto X'_j\}]}$$

where  $\eta_1$  is the only child of  $\eta$ ,  $e$  and  $e'$  are both introduced by  $\eta$  and  $e \sim e'$ , and both are labeled with nonterminal  $B$ .

The complexity of the parsing algorithm is again in  $O((3^d n)^{k+1})$ , where  $k$  is now the maximum treewidth of the dependency graph as defined in this section. In general, this treewidth will be greater than the treewidth of either the source or target side on its own, so that synchronous parsing is generally slower than standard parsing.

## 5 Conclusion

Although Lautemann's polynomial-time extension of CKY to HRGs has been known for some time, the desire to use graph grammars for large-scale NLP applications introduces some practical considerations not accounted for in Lautemann's original presentation. We have provided a detailed description of our refinement of his algorithm and its implementation. It runs in  $O((3^d n)^{k+1})$  time and requires  $O((2^d n)^{k+1})$  space, where  $n$  is the number of nodes in the input graph,  $d$  is its maximum degree, and  $k$  is the maximum treewidth of the rule right-hand sides in the grammar. We have also described how to extend this algorithm to synchronous parsing. The parsing algorithms described in this paper are implemented in the Bolinas toolkit.<sup>3</sup>

<sup>3</sup>The Bolinas toolkit can be downloaded from <http://www.isi.edu/licensed-sw/bolinas/>.

## Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research was supported in part by ARO grant W911NF-10-1-0533.

## References

- Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. 1987. Complexity of finding embeddings in a  $k$ -tree. *SIAM Journal on Algebraic and Discrete Methods*, 8(2).
- Hans L. Bodlaender. 1997. Treewidth: Algorithmic techniques and results. In *Proc. 22nd International Symposium on Mathematical Foundations of Computer Science (MFCS '97)*, pages 29–36, Berlin. Springer-Verlag.
- Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michał Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. 2011. Solving connectivity problems parameterized by treewidth in single exponential time. *Computing Research Repository*, abs/1103.0534.
- Frank Drewes, Hans-Jörg Kreowski, and Annegret Habel. 1997. Hyperedge replacement graph grammars. In Grzegorz Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation*, pages 95–162. World Scientific.
- Daniel Gildea. 2011. Grammar factorization by tree decomposition. *Computational Linguistics*, 37(1):231–248.
- Vibhav Gogate and Rina Dechter. 2004. A complete anytime algorithm for treewidth. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.
- Bevan Jones, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, and Kevin Knight. 2012. Semantics-based machine translation with hyperedge replacement grammars. In *Proc. COLING*.
- Aravind K. Joshi and Yves Schabes. 1997. Tree-adjointing grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages and Automata*, volume 3, pages 69–124. Springer.
- Clemens Lautemann. 1990. The complexity of graph languages generated by hyperedge replacement. *Acta Informatica*, 27:399–421.
- Steffen Mazanek and Mark Minas. 2008. Parsing of hyperedge replacement grammars with graph parser combinators. In *Proc. 7th International Workshop on Graph Transformation and Visual Modeling Techniques*.
- Richard Moot. 2008. Lambek grammars, tree adjoining grammars and hyperedge replacement grammars. In *Proc. TAG+9*, pages 65–72.
- Grzegorz Rozenberg and Emo Welzl. 1986. Boundary NLC graph grammars—basic definitions, normal forms, and complexity. *Information and Control*, 69:136–167.
- Stuart M. Shieber, Yves Schabes, and Fernando C. N. Pereira. 1995. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24:3–36.
- Lappoon Tang and Raymond Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proc. European Conference on Machine Learning*.

# Grounded Unsupervised Semantic Parsing

Hoifung Poon

One Microsoft Way  
Microsoft Research  
Redmond, WA 98052, USA

hoifung@microsoft.com

## Abstract

We present the first unsupervised approach for semantic parsing that rivals the accuracy of supervised approaches in translating natural-language questions to database queries. Our GUSP system produces a semantic parse by annotating the dependency-tree nodes and edges with latent states, and learns a probabilistic grammar using EM. To compensate for the lack of example annotations or question-answer pairs, GUSP adopts a novel grounded-learning approach to leverage database for indirect supervision. On the challenging ATIS dataset, GUSP attained an accuracy of 84%, effectively tying with the best published results by supervised approaches.

## 1 Introduction

Semantic parsing maps text to a formal meaning representation such as logical forms or structured queries. Recently, there has been a burgeoning interest in developing machine-learning approaches for semantic parsing (Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007; Mooney, 2007; Kwiatkowski et al., 2011), but the predominant paradigm uses supervised learning, which requires example annotations that are costly to obtain. More recently, several grounded-learning approaches have been proposed to alleviate the annotation burden (Chen and Mooney, 2008; Kim and Mooney, 2010; Börschinger et al., 2011; Clarke et al., 2010; Liang et al., 2011). In particular, Clarke et al. (2010) and Liang et al. (2011) proposed methods to learn from question-answer pairs alone, which represents a significant advance. However, although these methods exonerate annotators from mastering specialized logical forms, finding the answers for complex ques-

tions still requires non-trivial effort.<sup>1</sup>

Poon & Domingos (2009, 2010) proposed the USP system for unsupervised semantic parsing, which learns a parser by recursively clustering and composing synonymous expressions. While their approach completely obviates the need for direct supervision, their target logic forms are self-induced clusters, which do not align with existing database or ontology. As a result, USP can not be used directly to answer complex questions against an existing database. More importantly, it misses the opportunity to leverage database for indirect supervision.

In this paper, we present the GUSP system, which combines unsupervised semantic parsing with grounded learning from a database. GUSP starts with the dependency tree of a sentence and produces a semantic parse by annotating the nodes and edges with latent semantic states derived from the database. Given a set of natural-language questions and a database, GUSP learns a probabilistic semantic grammar using EM. To compensate for the lack of direct supervision, GUSP constrains the search space using the database schema, and bootstraps learning using lexical scores computed from the names and values of database elements.

Unlike previous grounded-learning approaches, GUSP does not require ambiguous annotations or oracle answers, but rather focuses on leveraging database contents that are readily available. Unlike USP, GUSP predetermines the target logical forms based on the database schema, which alleviates the difficulty in learning and ensures that the output semantic parses can be directly used in querying the database. To handle syntax-semantics mismatch, GUSP introduces a novel dependency-based meaning representation

---

<sup>1</sup>Clarke et al. (2010) and Liang et al. (2011) used the annotated logical forms to compute answers for their experiments.

by augmenting the state space to represent semantic relations beyond immediate dependency neighborhood. This representation also factorizes over nodes and edges, enabling linear-time exact inference in GUSP.

We evaluated GUSP on end-to-end question answering using the ATIS dataset for semantic parsing (Zettlemoyer and Collins, 2007). Compared to other standard datasets such as GEO and JOBS, ATIS features a database that is an order of magnitude larger in the numbers of relations and instances, as well as a more irregular language (ATIS questions were derived from spoken dialogs). Despite these challenges, GUSP attains an accuracy of 84% in end-to-end question answering, effectively tying with the state-of-the-art supervised approaches (85% by Zettlemoyer & Collins (2007), 83% by Kwiatkowski et al. (2011)).

## 2 Background

### 2.1 Semantic Parsing

The goal of semantic parsing is to map text to a complete and detailed meaning representation (Mooney, 2007). This is in contrast with semantic role labeling (Carreras and Marquez, 2004) and information extraction (Banko et al., 2007; Poon and Domingos, 2007), which have a more restricted goal of identifying local semantic roles or extracting selected information slots.

The standard language for meaning representation is first-order logic or a sublanguage, such as FunQL (Kate et al., 2005; Clarke et al., 2010) and lambda calculus (Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007). Poon & Domingos (2009, 2010) induce a meaning representation by clustering synonymous lambda-calculus forms stemming from partitions of dependency trees. More recently, Liang et al. (2011) proposed DCS for dependency-based compositional semantics, which represents a semantic parse as a tree with nodes representing database elements and operations, and edges representing relational joins.

In this paper, we focus on semantic parsing for natural-language interface to database (Grosz et al., 1987). In this problem setting, a natural-language question is first translated into a meaning representation by semantic parsing, and then converted into a structured query such as SQL to obtain answer from the database.

### 2.2 Unsupervised Semantic Parsing

Unsupervised semantic parsing was first proposed by Poon & Domingos (2009, 2010) with their USP system. USP defines a probabilistic model over the dependency tree and semantic parse using Markov logic (Domingos and Lowd, 2009), and recursively clusters and composes synonymous dependency treelets using a hard EM-like procedure. Since USP uses nonlocal features (e.g., the argument-number feature) and operates over partitions, exact inference is intractable, and USP resorts to a greedy approach to find the MAP parse by searching over partitions. Titov & Klementiev (2011) proposed a Bayesian version of USP and Titov & Klementiev (2012) adapted it for semantic role induction. In USP, the meaning is represented by self-induced clusters. Therefore, to answer complex questions against a database, it requires an additional ontology matching step to resolve USP clusters with database elements.

Popescu et al. (2003, 2004) proposed the PRECISE system, which does not require labeled examples and can be directly applied to question answering with a database. The PRECISE system, however, requires substantial amount of engineering, including a domain-specific lexicon that specifies the synonyms for names and values of database elements, a restricted set of potential interpretations for domain verbs and prepositions, as well as a set of domain questions with manually labeled POS tags for retraining the tagger and parser. It also focuses on the subset of easy questions (“semantically tractable” questions), and sidesteps the problem of dealing with complex and nested structures, as well as ambiguous interpretations. Remarkably, while PRECISE can be very accurate on easy questions, it does not try to learn from these interpretations. In contrast, Goldwasser et al. (2011) proposed a self-supervised approach, which iteratively chose high-confidence parses to retrain the parser. Their system, however, still required a lexicon manually constructed for the given domain. Moreover, it was only applied to a small domain (a subset of GEO), and the result still trailed supervised systems by a wide margin.

### 2.3 Grounded Learning for Semantic Parsing

Grounded learning is motivated by alleviating the burden of direct supervision via interaction with the world, where the indirect supervision may take the form as ambiguous annotations (Chen

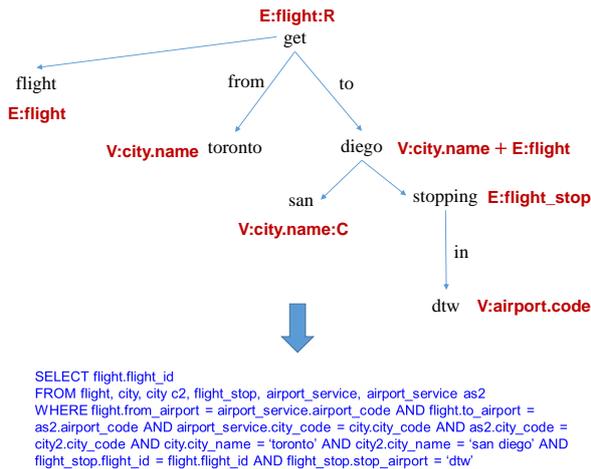


Figure 1: End-to-end question answering by GUSP for sentence *get flight from toronto to san diego stopping in dtw*. Top: the dependency tree of the sentence is annotated with latent semantic states by GUSP. For brevity, we omit the edge states. Raising occurs from *flight* to *get* and sinking occurs from *get* to *diego*. Bottom: the semantic tree is deterministically converted into SQL to obtain answer from the database.

and Mooney, 2008; Kim and Mooney, 2010; Börschinger et al., 2011) or example question-answer pairs (Clarke et al., 2010; Liang et al., 2011). In general, however, such supervision is not always available or easy to obtain. In contrast, databases are often abundantly available, especially for important domains.

The database community has considerable amount of work on leveraging databases in various tasks such as entity resolution, schema matching, and others. To the best of our knowledge, this approach is still underexplored in the NLP community. One notable exception is distant supervision (Mintz et al., 2009; Riedel et al., 2010; Hoffmann et al., 2011; Krishnamurthy and Mitchell, 2012; Heck et al., 2013), which used database instances to derive training examples for relation extraction. This approach, however, still has considerable limitations. For example, it only handles binary relations, and the quality of the training examples is inherently noisy and hard to control. Moreover, this approach is not applicable to the question-answering setting considered in this paper, since entity pairs in questions need not correspond to valid relational instances in the database.

### 3 Grounded Unsupervised Semantic Parsing

In this section, we present the GUSP system for grounded unsupervised semantic parsing. GUSP is unsupervised and does not require example logical forms or question-answer pairs. Figure 1 shows an example of end-to-end question answering using GUSP. GUSP produces a semantic parse of the question by annotating its dependency tree with latent semantic states. The semantic tree can then be deterministically converted into SQL to obtain answer from the database. Given a set of natural-language questions and a database, GUSP learns a probabilistic semantic grammar using EM.

To compensate for the lack of annotated examples, GUSP derives indirect supervision from a novel combination of three key sources. First, GUSP leverages the target database to constrain the search space. Specifically, it defines the semantic states based on the database schema, and derives lexical-trigger scores from database elements to bootstrap learning.

Second, in contrast to most existing approaches for semantic parsing, GUSP starts directly from dependency trees and focuses on translating them into semantic parses. While syntax may not always align perfectly with semantics, it is still highly informative about the latter. In particular, dependency edges are often indicative of semantic relations. On the other hand, syntax and semantic often diverge, and syntactic parsing errors abound. To combat this problem, GUSP introduces a novel dependency-based meaning representation with an augmented state space to account for semantic relations that are nonlocal in the dependency tree.

GUSP’s approach of starting directly from dependency tree is inspired by USP. However, GUSP uses a different meaning representation defined over individual nodes and edges, rather than partitions, which enables linear-time exact inference. GUSP also handles complex linguistic phenomena and syntax-semantics mismatch by explicitly augmenting the state space, whereas USP’s capability in handling such phenomena is indirect and more limited.

GUSP represents meaning by a semantic tree, which is similar to DCS (Liang et al., 2011). Their approach to semantic parsing, however, differs from GUSP in that it induced the semantic tree directly from a sentence, rather than starting from

a dependency tree and annotating it. Their approach alleviates some complexity in the meaning representation for handling syntax-semantics mismatch, but it has to search over a much larger search space involving exponentially many candidate trees. This might partially explain why it has not yet been scaled up to the ATIS dataset.

Finally, GUSP recognizes that certain aspects in semantic parsing may not be worth learning using precious annotated examples. These are domain-independent and closed-class expressions, such as times and dates (e.g., *before 5pm* and *July seventeenth*), logical connectives (e.g., *and*, *or*, *not*), and numerics (e.g., *200 dollars*). GUSP preprocesses the text to detect such expressions and restricts their interpretation to database elements of compatible types (e.g., *before 5pm* vs. `flight.departure_time` or `flight.arrival_time`). Short of training examples, GUSP also resolves quantifier scoping ambiguities deterministically by a fixed ordering. For example, in the phrase *cheapest flight to Seattle*, the scope of *cheapest* can be either *flight* or *flight to seattle*. GUSP always chooses to apply the superlative at last, amounting to choosing the most restricted scope (*flight to seattle*), which is usually the correct interpretation.

In the remainder of this section, we first formalize the problem setting and introduce the GUSP meaning representation. We then present the GUSP model and learning and inference algorithms. Finally, we describe how to convert a GUSP semantic parse into SQL.

### 3.1 Problem Formulation

Let  $d$  be a dependency tree,  $N(d)$  and  $E(d)$  be its nodes and edges. In GUSP, a semantic parse of  $d$  is an assignment  $z : N(d) \cup E(d) \rightarrow \mathcal{S}$  that maps its nodes and edges to semantic states in  $\mathcal{S}$ . For example, in the example in Figure 1,  $z(\textit{flight}) = E:\textit{flight}$ . At the core of GUSP is a joint probability distribution  $P_\theta(d, z)$  over the dependency tree and the semantic parse. Semantic parsing in GUSP amounts to finding the most probable parse  $z^* = \arg \max_z P_\theta(d, z)$ . Given a set of sentences and their dependency trees  $D$ , learning in GUSP maximizes the log-likelihood of  $D$  while summing out the latent parses  $z$ :

$$\begin{aligned} \theta^* &= \arg \max \log P_\theta(D) \\ &= \arg \max \sum_{d \in D} \log \sum_z P_\theta(d, z) \end{aligned}$$

### 3.2 Simple Semantic States

**Node states** GUSP creates a state  $E:X$  ( $E$  short for entity) for each database entity  $X$  (i.e., a database table), a state  $P:Y$  ( $P$  short for property) and  $V:Y$  ( $V$  short for value) for each database attribute  $Y$  (i.e., a database column). Node states are assigned to dependency nodes. Intuitively, they represent database entities, properties, and values. For example, the ATIS domain contains entities such as `flight` and `fare`, which may contain properties such as the departure time `flight.departure_time` or ticket price `fare.one_direction_cost`. The mentions of entities and properties are represented by entity and property states, whereas constants such as `9:25am` or `120 dollars` are represented by value states. In the semantic parse in Figure 1, for example, *flight* is assigned to entity state  $E:\textit{flight}$ , where *toronto* is assigned to value state  $V:\textit{city.name}$ . There is a special node state `NULL`, which signifies that the subtree headed by the word contributes no meaning to the semantic parse (e.g., an auxiliary verb).

**Edge states** GUSP creates an edge state for each valid relational join paths connecting two node states. Edge states are assigned to dependency edges. GUSP enforces the constraints that the node states of the dependency parent and child must agree with the node states in the edge state. For example,  $E:\textit{flight}-V:\textit{flight.departure_time}$  represents a natural join between the flight entity and the property value departure time. For a dependency edge  $e : a \rightarrow b$ , the assignment to  $E:\textit{flight}-V:\textit{flight.departure_time}$  signifies that  $a$  represents a flight entity, and  $b$  represents the value of its departure time. An edge state may also represent a relational path consisting of a serial of joins. For example, Zettlemoyer and Collins (2007) used a predicate `from(f, c)` to signify that flight  $f$  starts from city  $c$ . In the ATIS database, however, this amounts to a path of three joins:

```
flight.from-airport-airport
airport-airport-service
airport-service-city
```

In GUSP, this is represented by the edge state `flight-flight.from-airport-airport-service-city`.

GUSP only creates edge states for relational join paths up to length four, as longer paths rarely correspond to meaningful semantic relations.

**Composition** To handle compositions such as *American Airlines* and *New York City*, it helps to distinguish the head words (*Airlines* and *City*) from the rest. In GUSP, this is handled by introducing, for each node state such as `E:airline`, a new node state such as `E:airline:C`, where `C` signifies composition. For example, in Figure 1, *diego* is assigned to `V:city.name`, whereas *san diego* is assigned to `V:city.name:C`, since *san diego* forms a single meaning unit, and should be translated into SQL as a whole.

### 3.3 Domain-Independent States

These are for handling special linguistic phenomena that are not domain-specific, such as negation, superlatives, and quantifiers.

**Operator states** GUSP create node states for the logical and comparison operators (`OR`, `AND`, `NOT`, `MORE`, `LESS`, `EQ`). Additionally, to handle the cases when prepositions and logical connectives are collapsed into the label of a dependency edge, as in Stanford dependency, GUSP introduces an edge state for each triple of an operator and two node states, such as `E:flight-AND-E:fare`.

**Quantifier states** GUSP creates a node state for each of the standard SQL functions: `argmin`, `argmax`, `count`, `sum`. Additionally, it creates a node state for each pair of compatible function and property. For example, `argmin` can be applied to any numeric property, in particular `flight.departure_time`, and so the node state `P:flight.departure_time:argmin` is created and can be assigned to superlatives such as *earliest*.

### 3.4 Complex Semantic States

For sentences with a correct dependency tree and well-aligned syntax and semantics, the simple semantic states suffice for annotating the correct semantic parse. However, in complex sentences, syntax and semantic often diverge, either due to their differing goals or simply stemming from syntactic parsing errors. In Figure 1, the dependency tree contains multiple errors: *from toronto* and *to san diego* are mistakenly attached to *get*, which has no literal meaning here; *stopping in dtw* is also

wrongly attached to *diego* rather than *flight*. Annotating such a tree with only simple states will lead to incorrect semantic parses, e.g., by joining `V:city:san diego` with `V:airport:dtw` via `E:airport_service`, rather than joining `E:flight` with `V:airport:dtw` via `E:flight_stop`.

To overcome these challenges, GUSP introduces three types of complex states to handle syntax-semantics divergence. Figure 1 shows the correct semantic parse for the above sentence using the complex states.

**Raising** For each simple node state `N`, GUSP creates a “raised” state `N:R` (`R` short for raised). A raised state signifies a word that has little or none of its own meaning, but effectively takes one of its child states to be its own (“raises”). Correspondingly, GUSP creates a “raising” edge state `N-R-N`, which signifies that the parent is a raised state and its meaning is derived from the dependency child of state `N`. For all other children, the parent behaves just as state `N`. For example, in Figure 1, *get* is assigned to the raised state `E:flight:R`, and the edge between *get* and *flight* is assigned to the raising edge state `E:flight-R-E:flight`.

**Sinking** For simple node states `A`, `B` and an edge state `E` connecting the two, GUSP creates a “sinking” node state `A+E+B:S` (`S` for sinking). When a node `n` is assigned to such a sinking state, `n` can behave as either `A` or `B` for its children (i.e., the edge states can connect to either one), and `n`’s parent must be of state `B`. In Figure 1, for example, *diego* is assigned to a sinking state `V:city.name + E:flight` (the edge state is omitted for brevity). `E:flight` comes from its parent *get*. For child *san diego* behaves as in state `V:city.name`, and their edge state is a simple compositional join. For the other child *stopping diego* behaves as in state `E:flight`, and their edge state is a relational join connecting *flight* with *flight\_stop*. Effectively, this connects *stopping* with *get* and eventually with *flight* (due to raising), virtually correcting the syntax-semantics mismatch stemming from attachment errors.

**Implicit** For simple node states `A`, `B` and an edge state `E` connecting the two, GUSP also creates a node state `A+E+B:I` (`I` for implicit) with the “implicit” state `B`. In natural languages, an entity is often introduced implicitly, which the reader infers from shared world knowledge. For example,

to obtain the correct semantic parse for *Give me the fare from Seattle to Boston*, one needs to infer the existence of a *flight* entity, as in *Give me the fare (of a flight) from Seattle to Boston*. Implicit states offer candidates for addressing such needs. As in sinking, child nodes have access to either of the two simple states, but the implicit state is not visible to the parent node.

### 3.5 Lexical-Trigger Scores

GUSP uses the database elements to automatically derive a simple scoring scheme for lexical triggers. If a database element has a name of  $k$  words, each word is assigned score  $1/k$  for the corresponding node state. Similarly for property values and value node states. In a sentence, if a word  $w$  triggers a node state with score  $s$ , its dependency children and left and right neighbors all get a trigger score of  $0.1 \cdot s$  for the same state. To score relevant words not appearing in the database (due to incompleteness of the database or lexical variations), GUSP uses DASH (Pantel et al., 2009) to provide additional word-pair scoring based on lexical distributional similarity computed over general text corpora (Wikipedia in this case). In the case of multiple score assignments for the same word, the maximum score is used.

For multi-word values of property  $Y$ , and for a dependency edge connecting two collocated words, GUSP assigns a score 1.0 to the edge state joining the value node state  $V:Y$  to its composition state  $V:Y:C$ , as well as the edge state joining two composition states  $V:Y:C$ .

GUSP also uses a domain-independent list of superlatives with the corresponding data types and polarity (e.g., *first*, *last*, *earliest*, *latest*, *cheapest*) and assigns a trigger score of 1.0 for each property of a compatible data type (e.g., *cheapest* for properties of type MONEY).

### 3.6 The GUSP Model

In a nutshell, the GUSP model resembles a tree-HMM, which models the emission of words and dependencies by node and edge states, as well as transition between an edge state and the parent and child node states. In preliminary experiments on the development set, we found that the naïve model (with multinomials as conditional probabilities) did not perform well in EM. We thus chose to apply feature-rich EM (Berg-Kirkpatrick et al., 2010) in GUSP, which enabled the use of more generalizable features. Specifically, GUSP defines

a probability distribution over dependency tree  $d$  and semantic parse  $z$  by

$$P_{\theta}(d, z) = \frac{1}{Z} \exp \sum_i f_i(d, z) \cdot w_i(d, z)$$

where  $f_i$  and  $w_i$  are features and their weights, and  $Z$  is the normalization constant that sums over all possible  $d, z$  (over the same unlabeled tree). The features of GUSP are as follows:

**Lexical-trigger scores** These are implemented as emission features with fixed weights. For example, given a token  $t$  that triggers node state  $N$  with score  $s$ , there is a corresponding features  $1(\text{lemma} = t, \text{state} = N)$  with weight  $\alpha \cdot s$ , where  $\alpha$  is a parameter.

**Emission features for node states** GUSP uses two templates for emission of node states: for raised states,  $1(\text{token} = \cdot)$ , i.e., the emission weights for all raised states are tied; for non-raised states,  $1(\text{lemma} = \cdot, \text{state} = N)$ .

**Emission features for edge states** GUSP uses the following templates for emission of edge states:

Child node state is NULL, dependency=  $\cdot$ ;

Edge state is RAISING, dependency=  $\cdot$ ;

Parent node state is same as the child node state, dependency=  $\cdot$ ;

Otherwise, parent node state=  $\cdot$ , child node state=  $\cdot$ , edge state type=  $\cdot$ , dependency=  $\cdot$ .

**Transition features** GUSP uses the following templates for transition features, which are similar to the edge emission features except for the dependency label:

Child node state is NULL;

Edge state is RAISING;

Parent node state is same as the child node state;

Otherwise, parent node state=  $\cdot$ , child node state=  $\cdot$ , edge state type=  $\cdot$ .

**Complexity Prior** To favor simple semantic parses, GUSP imposes an exponential prior with weight  $\beta$  on nodes states that are not null or raised, and on each relational join in an edge state.

### 3.7 Learning and Inference

Since the GUSP model factors over nodes and edges, learning and inference can be done efficiently using EM and dynamic programming. Specifically, the MAP parse and expectations can

be computed by tree-Viterbi and inside-outside (Petrov and Klein, 2008). The parameters can be estimated by feature-rich EM (Berg-Kirkpatrick et al., 2010).

Because the Viterbi and inside-outside are applied to a fixed tree (i.e., the input dependency tree), their running times are only linear in the sentence length in GUSP.

### 3.8 Query Generation

Given a semantic parse, GUSP generates the SQL by a depth-first traversal that recursively computes the denotation of a node from the denotations of its children and its node state and edge states. Each denotation is a structured query that contains: a list of entities for projection (corresponding to the FROM statement in SQL); a computation tree where the leaves are simple joins or value comparisons, and the internal nodes are logical or quantifier operators (the WHERE statement); the salient database elements (the SELECT statement). Below, we illustrate this procedure using the semantic parse in Figure 1 as a running example.

**Value node state** GUSP creates a semantic object of the given type with a unique index and the word constant. For example, the denotation for node *toronto* is a `city.name` object with a unique index and constant “toronto”. The unique index is necessary in case the SQL involves multiple instances of the same entity. For example, the SQL in Figure 1 involves two instances of the entity `city`, corresponding to the departure and arrival cities, respectively. By default, such a semantic object will be translated into an equality constraint, such as `city.name = toronto`.

**Entity or property node state** GUSP creates a semantic object of the given type with a unique relation index. For example, the denotation for node *flight* is simply a `flight` object with a unique index. By default, such an object will contribute to the list of entities in SQL projection (the FROM statement), but not any constraints.

**NULL state** GUSP returns an empty denotation.

**Simple edge state** GUSP appends the child denotation to that of the parent, and appends equality constraints corresponding to the relational join path. In the case of composition, such as the join between *diego* and *san*, GUSP simply keeps the parent object, while adding to it the words from

the child. In the case of a more complex join, such as that between *stopping* and *dtw*, GUSP adds the relational constraints that join `flight_stop` with `airport`:

`flight_stop.stop_airport = airport.airport_id`.

**Raising edge state** GUSP simply takes the child denotation and sets that to the parent.

**Implicit and sinking states** GUSP maintains two separate denotations for the two simple states in the complex state, and processes their respective edge states accordingly. For example, the node *diego* contains two denotations, one for `V:city.name`, and one for `E:flight`, with the corresponding child being *san* and *stopping*, respectively.

**Domain-independent states** For comparator states such as `MORE` or `LESS`, GUSP changes the default equality constraints to an inequality one, such as `flight.depart_time < 600` for *before 6am*. For logical connectives, GUSP combines the projection and constraints accordingly. For quantifier states, GUSP applies the given function to the query.

**Resolve scoping ambiguities** GUSP delays applying quantifiers until the child semantic object differs from the parent one or when reaching the root. GUSP employs the following fixed ordering in evaluating quantifiers and operators: superlatives and other quantifiers are evaluated at last (i.e., after evaluating all other joins or operators for the given object), whereas negation is evaluated first, conjunctions and disjunctions are evaluated in their order of appearance.

## 4 Experiments

### 4.1 Task

We evaluated GUSP on the ATIS travel planning domain, which has been studied in He & Young (2005, 2006) and adapted for evaluating semantic parsing by Zettlemoyer & Collins (2007) (henceforth ZC07). The ZC07 dataset contains annotated logical forms for each sentence, which we do not use. Since our goal is not to produce a specific logical form, we directly evaluate on the end-to-end task of translating questions into database queries and measure question-answering accuracy. The ATIS distribution contains the original SQL annotations, which we used to compute gold answers

for evaluation only. The dataset is split into training, development, and test, containing 4500, 478, and 449 sentences, respectively. We used the development set for initial development and tuning hyperparameters. At test time, we ran GUSP over the test set to learn a semantic parser and output the MAP parses.<sup>2</sup>

## 4.2 Preprocessing

The ATIS sentences were originally derived from spoken dialog and were therefore in lower cases. Since case information is important for parsers and taggers, we first truecased the sentences using DASH (Pantel et al., 2009), which stores the case for each phrase in Wikipedia.

We then ran the sentences through SPLAT, a state-of-the-art NLP toolkit (Quirk et al., 2012), to conduct tokenization, part-of-speech tagging, and constituency parsing. Since SPLAT does not output dependency trees, we ran the Stanford parser over SPLAT parses to generate the dependency trees in Stanford dependency (de Marneffe et al., 2006).

## 4.3 Systems

For the GUSP system, we set the hyperparameters from initial experiments on the development set, and used them in all subsequent experiments. Specifically, we set  $\alpha = 50$  and  $\beta = -0.1$ , and ran three iterations of feature-rich EM with an  $L_2$  prior of 10 over the feature weights.

To evaluate the importance of complex states, we considered two versions of GUSP : **GUSP-SIMPLE** and **GUSP-FULL**, where **GUSP-SIMPLE** only admits simple states, whereas **GUSP-FULL** admits all states.

During development, we found that some questions are inherently ambiguous that cannot be solved except with some domain knowledge or labeled examples. In Section 3.2, we discuss an edge state that joins a flight with its starting city: `flight-flight.from.airport-airport-airport.service-city`. The ATIS database also contains another path of the same length: `flight-flight.from.airport-airport-ground.service-city`. The only difference is that `air.service` is replaced by `ground.service`. In some occasions, the

<sup>2</sup>This doesn't lead to overfitting since we did not use any labeled information in the test set.

Table 1: Comparison of semantic parsing accuracy on the ATIS test dataset. Both ZC07 and FUBL used annotated logical forms in training, whereas GUSP-FULL and GUSP++ did not. The numbers for GUSP-FULL and GUSP++ are end-to-end question answering accuracy, whereas the numbers for ZC07 and FUBL are recall on exact match in logical forms.

	Accuracy
ZC07	84.6
FUBL	82.8
GUSP-FULL	74.8
GUSP++	83.5

answers are identical whereas in others they are different. Without other information, neither the complexity prior nor EM can properly discriminate one against another. (Note that this ambiguity is not present in the ZC07 logical forms, which use a single predicate `from(f, c)` for the entire relation paths. In other words, to translate ZC07 logical forms into SQL, one also needs to decide on which path to use.)

Another type of domain-specific ambiguities involves sentences such as *give me information on flights after 4pm on wednesday*. There is no obvious information to disambiguate between `flight.departure.time` and `flight.arrival.time` for *4pm*.

Such ambiguities suggest opportunities for interactive learning,<sup>3</sup> but this is clearly out of the scope of this paper. Instead, we incorporated a simple disambiguation feature with a small weight of 0.01 that fires over the simple states of `flight.departure.time` and `airport.service`. We named the resulting system **GUSP++**.

To gauge the difficulty of the task and the quality of lexical-trigger scores, we also considered a deterministic baseline **LEXICAL**, which computed semantic parses using lexical-trigger scores alone.

<sup>3</sup>For example, after eliminating other much less likely alternatives, the system can present to the user with both choices and let the user to choose the correct one. The implicit feedback signal can then be used to train the system for future disambiguation.

Table 2: Comparison of question answering accuracy in ablation experiments.

	Accuracy
LEXICAL	33.9
GUSP-SIMPLE	66.5
GUSP-FULL	74.8
GUSP++	83.5
– RAISING	75.7
– SINKING	77.5
– IMPLICIT	76.2

#### 4.4 Results

We first compared the results of GUSP-FULL and GUSP++ with ZC07 and FUBL (Kwiatkowski et al., 2011).<sup>4</sup> Note that ZC07 and FUBL were evaluated on exact match in logical forms. We used their recall numbers which are the percentages of sentences with fully correct logical forms. Given that the questions are quite specific and generally admit nonzero number of answers, the question-answer accuracy should be quite comparable with these numbers.

Table 1 shows the comparison. Surprisingly, even without the additional disambiguation feature, GUSP-FULL already attained an accuracy broadly in range with supervised results. With the feature, GUSP++ effectively tied with the best supervised approach.

To evaluate the importance of various components in GUSP, we conducted ablation test to compare the variants of GUSP. Table 2 shows the results. LEXICAL can parse more than one third of the sentences correctly, which is quite remarkable in itself, considering that it only used the lexical scores. On the other hand, roughly two-third of the sentences cannot be correctly parsed in this way, suggesting that the lexical scores are noisy and ambiguous. In comparison, all GUSP variants achieved significant gains over LEXICAL. Additionally, GUSP-FULL substantially outperformed GUSP-SIMPLE, highlighting the challenges of syntax-semantics mismatch in ATIS, and demonstrating the importance and effectiveness of complex states for handling such mismatch. All three types of complex states produced significant contributions. For example, compared to GUSP++,

<sup>4</sup>We should note that while the more recent system of FUBL slightly trails ZC07, it is language-independent and can parse questions in multiple languages.

removing RAISING dropped accuracy by almost 8 points.

#### 4.5 Discussion

Upon manual inspection, many of the remaining errors are due to syntactic parsing errors that are too severe to fix. This is partly due to the fact that ATIS sentences are out of domain compared to the newswired text on which the syntactic parsers were trained. For example, *show*, *list* were regularly parsed as nouns, whereas *round* (as in *round trip*) were often parsed as a verb and *northwest* were parsed as an auxiliary verb. Another reason is that ATIS sentences are typically less formal or grammatical, which exacerbates the difficulty in parsing. In this paper, we used the 1-best dependency tree to produce semantic parse. An interesting future direction is to consider joint syntactic-semantic parsing, using  $k$ -best trees or even the parse forest as input and reranking the top parse using semantic information.<sup>5</sup>

#### 5 Conclusion

This paper introduces grounded unsupervised semantic parsing, which leverages available database for indirect supervision and uses a grounded meaning representation to account for syntax-semantics mismatch in dependency-based semantic parsing. The resulting GUSP system is the first unsupervised approach to attain an accuracy comparable to the best supervised systems in translating complex natural-language questions to database queries.

Directions for future work include: joint syntactic-semantic parsing, developing better features for learning; interactive learning in a dialog setting; generalizing distant supervision; application to knowledge extraction from database-rich domains such as biomedical sciences.

#### Acknowledgments

We would like to thank Kristina Toutanova, Chris Quirk, Luke Zettlemoyer, and Yoav Artzi for useful discussions, and Patrick Pantel and Michael Gammon for help with the datasets.

<sup>5</sup>Note that this is still different from the currently predominant approaches in semantic parsing, which learn to parse both syntax and semantics by training from the *semantic parsing datasets* alone, which are considerably smaller compared to resources available for syntactic parsing.

## References

- Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 2670–2676, Hyderabad, India. AAAI Press.
- Taylor Berg-Kirkpatrick, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.
- Benjamin Börschinger, Bevan K. Jones, and Mark Johnson. 2011. Reducing grounded learning tasks to grammatical inference. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Xavier Carreras and Luis Marquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning*, pages 89–97, Boston, MA. ACL.
- David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *ICML-08*.
- James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from world’s response. In *Proceedings of the 2010 Conference on Natural Language Learning*.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 449–454, Genoa, Italy. ELRA.
- Pedro Domingos and Daniel Lowd. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool, San Rafael, CA.
- Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of the Forty Ninth Annual Meeting of the Association for Computational Linguistics*.
- B.J. Grosz, D. Appelt, P. Martin, and F. Pereira. 1987. Team: An experiment in the design of transportable natural language interfaces. *Artificial Intelligence*, 32:173–243.
- Yulan He and Steve Young. 2005. Semantic processing using the hidden vector state model. In *Computer Speech and Language*.
- Yulan He and Steve Young. 2006. Spoken language understanding using the hidden vector state model. In *Speech Communication Special Issue on Spoken Language understanding for Conversational Systems*.
- Larry Heck, Dilek Hakkani-Tur, and Gokhan Tur. 2013. Leveraging knowledge graphs for web-scale unsupervised semantic parsing. In *Proceedings of the Interspeech 2013*.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the Forty Ninth Annual Meeting of the Association for Computational Linguistics*.
- R. J. Kate, Y. W. Wong, and R. J. Mooney. 2005. Learning to transform natural to formal languages. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*.
- Joohyun Kim and Raymond J. Mooney. 2010. Generative alignment and semantic parsing for learning from ambiguous supervision. In *COLING10*.
- Jayant Krishnamurthy and Tom M. Mitchell. 2012. Weakly supervised training of semantic parsers. In *EMNLP-12*.
- Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the Forty Ninth Annual Meeting of the Association for Computational Linguistics*.
- Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Forty Seventh Annual Meeting of the Association for Computational Linguistics*.
- Raymond J. Mooney. 2007. Learning for semantic parsing. In *Proceedings of the Eighth International Conference on Computational Linguistics and Intelligent Text Processing*, pages 311–324, Mexico City, Mexico. Springer.
- Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.
- Slav Petrov and Dan Klein. 2008. Discriminative log-linear grammars with latent variables. In *NIPS-08*.
- Hoifung Poon and Pedro Domingos. 2007. Joint inference in information extraction. In *Proceedings of the Twenty Second National Conference on Artificial Intelligence*, pages 913–918, Vancouver, Canada. AAAI Press.

- Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Singapore. ACL.
- Hoifung Poon and Pedro Domingos. 2010. Unsupervised ontological induction from text. In *Proceedings of the Forty Eighth Annual Meeting of the Association for Computational Linguistics*, pages 296–305, Uppsala, Sweden. ACL.
- Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. 2003. Towards a theory of natural language interfaces to databases. In *IUI-03*.
- Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates. 2004. Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In *COLING-04*.
- Chris Quirk, Pallavi Choudhury, Jianfeng Gao, Hisami Suzuki, Kristina Toutanova, Michael Gamon, Wentau Yih, and Lucy Vanderwende. 2012. MSR SPLAT, a language analysis toolkit. In *Proceedings of NAACL HLT 2012 Demonstration Session*.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the Sixteen European Conference on Machine Learning*.
- Ivan Titov and Alexandre Klementiev. 2011. A bayesian model for unsupervised semantic parsing. In *Proceedings of the Forty Ninth Annual Meeting of the Association for Computational Linguistics*.
- Ivan Titov and Alexandre Klementiev. 2012. A bayesian approach to unsupervised semantic role induction. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*.
- Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty First Conference on Uncertainty in Artificial Intelligence*, pages 658–666, Edinburgh, Scotland. AUAI Press.
- Luke S. Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

# Automatic detection of deception in child-produced speech using syntactic complexity features

**Maria Yancheva**

Division of Engineering Science,  
University of Toronto  
Toronto Ontario Canada  
maria.yancheva@utoronto.ca

**Frank Rudzicz**

Toronto Rehabilitation Institute; and  
Department of Computer Science,  
University of Toronto  
Toronto Ontario Canada  
frank@cs.toronto.edu

## Abstract

It is important that the testimony of children be admissible in court, especially given allegations of abuse. Unfortunately, children can be misled by interrogators or might offer false information, with dire consequences. In this work, we evaluate various parameterizations of five classifiers (including support vector machines, neural networks, and random forests) in deciphering truth from lies given transcripts of interviews with 198 victims of abuse between the ages of 4 and 7. These evaluations are performed using a novel set of syntactic features, including measures of complexity. Our results show that sentence length, the mean number of clauses per utterance, and the Stajner-Mitkov measure of complexity are highly informative syntactic features, that classification accuracy varies greatly by the age of the speaker, and that accuracy up to 91.7% can be achieved by support vector machines given a sufficient amount of data.

## 1 Introduction

The challenge of disambiguating between truth and deception is critical in determining the admissibility of court testimony. Unfortunately, the testimony of maltreated children is often not admitted in court due to concerns about truthfulness since children can be instructed to deny transgressions or misled to elicit false accusations (Lyon and Dorado, 2008). However, the child is often the only witness of the transgression (Undeutsch, 2008); automatically determining truthfulness in

such situations is therefore a paramount goal so that justice may be served effectively.

## 2 Related Work

Research in the detection of deception in adult speech has included analyses of verbal and non-verbal cues such as behavioral changes, facial expression, speech dysfluencies, and cognitive complexity (DePaulo et al., 2003). Despite statistically significant predictors of deception such as shorter talking time, fewer semantic details, and less coherent statements, DePaulo et al. (2003) found that the median effect size is very small. Deception without special motivation (e.g., everyday ‘white lies’) exhibited almost no discernible cues of deception. However, analysis of moderating factors showed that cues were significantly more numerous and salient when lies were about transgressions.

Literature on deception in children is relatively limited. In one study, Lewis et al. (1989) studied 3-year-olds and measured behavioral cues, such as facial expression and nervous body movement, before and after the elicitation of a lie. Verbal responses consisted of yes/no answers. Results suggested that 3-year-old children are capable of deception, and that non-verbal behaviors during deception include increases in ‘positive’ behaviors (e.g., smiling). However, verbal cues of deception were not analyzed. Crucially, Lewis et al. (1989) showed that humans are no more accurate in deciphering truth from deception in child speech than in adult speech, being only about 50% accurate.

More recently, researchers have used linguistic features to identify deception. Newman et al. (2003) inferred deception in transcribed, typed, and handwritten text by identifying features of linguistic style such as the use of personal pronouns

and exclusive words (e.g., *but*, *except*, *without*). These features were obtained with the Linguistic Inquiry and Word Count (LIWC) tool and used in a logistic regression classifier which achieved, on average, 61% accuracy on test data. Feature analysis showed that deceptive stories were characterized by fewer self-references, more negative emotion words, and lower cognitive complexity, compared to non-deceptive language.

Another recent stylistometric experiment in automatic identification of deception was performed by Mihalcea and Strapparava (2009). The authors used a dataset of truthful and deceptive typed responses produced by adult subjects on three different topics, collected through the Amazon Mechanical Turk service. Two classifiers, Naïve Bayes (NB) and a support vector machine (SVM), were applied on the tokenized and stemmed statements to obtain best classification accuracies of 70% (abortion topic, NB), 67.4% (death penalty topic, NB), and 77% (friend description, SVM), where the baseline was taken to be 50%. The large variability of classifier performance based on the topic of deception suggests that performance is context-dependent. The authors note this as well by demonstrating significantly lower results of 59.8% for NB and 57.8% for SVM when cross-topic classification is performed by training each classifier on two topics and testing on the third.

The Mihalcea-Strapparava mturk dataset was further used in a study by Feng et al. (2012) which employs lexicalized and unlexicalized production rules to obtain deep syntactic features. The cross-validation accuracy obtained on the three topics was improved to 77% (abortion topic), 71.5% (death penalty topic), and 85% (friend description). The results nevertheless varied with topic.

Another experiment using syntactic features for identifying sentences containing uncertain or unreliable information was conducted by Zheng et al. (2010) on an adult-produced dataset of abstracts and full articles from BioScope, and on paragraphs from Wikipedia. The results demonstrated that using syntactic dependency features extracted with the Stanford parser improved performance on the biological dataset, while an ensemble classifier combining a conditional random field (CRF) and a MaxEnt classifier performed better than individual classifiers on the Wikipedia dataset.

A meta-analysis of features used in deception detection was performed by Hauch et al. (2012)

and revealed that verbal cues based on lexical categories extracted using the LIWC tool show statistically significant, though small, differences between truth- and lie-tellers. Vartapetian and Gillam (2012) surveyed existing cues to verbal deception and demonstrated that features in LIWC are not indicative of deception in online content, recommending that the features used to identify deception and the thresholds between deception and truth be based on the specific data set.

In the speech community, analysis of deceptive speech has combined various acoustic, prosodic, and lexical features (Hirschberg et al., 2005). Gra-ciarena et al. (2006) combined two independent systems — an acoustic Gaussian mixture model based on Mel cepstral features, and a prosodic support vector machine based on features such as pitch, energy, and duration — and achieved an accuracy of 64.4% on a test subset of the Columbia-SRI-Colorado (CSC) corpus of deceptive and non-deceptive speech (Hirschberg et al., 2005).

While previous studies have achieved some promising results in detecting deception with lexical, acoustic, and prosodic features, syntax remains relatively unexplored compared to LIWC-based features. Syntactic complexity as a cue to deception is consistent with literature in social psychology which suggests that emotion suppression (e.g., inhibition of guilt and fear) consumes cognitive resources, which can influence the underlying complexity of utterances (Richards and Gross, 1999; Richards and Gross, 2000). Additionally, the use of syntactic features is motivated by their successful use on adult-produced datasets for detecting deceptive or uncertain utterances (Feng et al., 2012; Zheng et al., 2010), as well as in other applications, such as the evaluation of changes in text complexity (Stajner and Mitkov, 2012), the identification of personality in conversation and text (Mairesse et al., 2007), and the detection of dementia through syntactic changes in writing (Le et al., 2011).

Past work has focused on identifying deceptive speech produced by adults. The problem of determining validity of child testimony in high-stakes child abuse court cases motivates the analysis of child-produced deceptive language. Further, the use of binary classification schemes in previous work does not account for partial truths often encountered in real-life scenarios. Due to the rarity of real deceptive data, studies typically use arti-

ficially produced deceptive language which falls unambiguously in one of two classes: complete truth or complete deception (Newman et al., 2003; Mihalcea and Strapparava, 2009). Studies which make use of real high-stakes courtroom data containing partial truths, such as the Italian DECOUR corpus analyzed by Fornaciari and Poesio (2012), preprocess the dataset to eliminate any partially truthful utterances. Since utterances of this kind are common in real language, their elimination from the dataset is not ideal.

The present study evaluates the viability of a novel set of 17 syntactic features as markers of deception in five classifiers. Moreover, to our knowledge, it is the first application of automatic deception detection to a real-life dataset of deceptive speech produced by maltreated children. The data is scored using a gradient of truthfulness, which is used to represent completely true, partially true, and completely false statements. Descriptions of the data (section 3) and feature sets (section 4) precede experimental results (section 5) and the concluding discussion (section 6).

### 3 Data

The data used in this study were obtained from Lyon et al. (2008), who conducted and transcribed a truth-induction experiment involving maltreated children awaiting court appearances in the Los Angeles County Dependency Court. Subjects were children between the ages of 4 and 7 (99 boys and 99 girls) who were interviewed regarding an unambiguous minor transgression involving playing with a toy. To ensure an understanding of lying and its negative consequences, all children passed a preliminary oath-taking competency task, requiring each child to correctly identify a truth-teller and a lie-teller in an object labeling task, as well as to identify which of the two would be the target of negative consequences.

During data collection, a confederate first engaged each child individually in one of four conditions: a) *play*, b) *play and coach*, c) *no play*, and d) *no play and coach*. In the two *play* conditions, the confederate engaged the child in play with a toy house (in the *no play* conditions, they did not); in the two *coach* conditions, the confederate coached the child to lie (i.e., to deny playing if they played with the toy house, or to admit playing if they did not). The confederate then left and the child was interviewed by a second researcher who performed a truth-induction manipulation consisting

of one of: a) *control* — no manipulation, b) *oath* — the interviewer reminded the child of the importance of telling the truth and elicited a promise of truth-telling, and c) *reassurance* — the interviewer reassured the child that telling the truth will not lead to any negative consequences.

Each pre- and post-induction transcription may contain explicit statements of up to seven features: looking at toy-house, touching toy-house, playing with toy-house, opening toy-house doors or windows to uncover hidden toys, playing with these hidden toys, spinning the toy-house, and putting back or hiding a toy. All children in the *play* condition engaged in all seven actions, while children in the *no play* condition engaged in none. An eighth feature is the lack of explicit denial of touching or playing with the toy house, which is considered to be truthful in the *play* condition, and deceptive in the *no play* condition (see the examples in the appendix). A transcription is labeled as *truth* if at least half of these features are truthful (53.2% of all transcriptions) and *lie* otherwise (46.8% of transcriptions). Other thresholds for this binary discrimination are explored in section 5.4.

Each child's verbal response was recorded twice: at time  $T_1$  (prior to truth-induction), and at time  $T_2$  (after truth-induction). Each child was subject to one of the four confederate conditions and one of the three induction conditions. The raw data were pre-processed to remove subjects with blank transcriptions, resulting in a total of 173 subjects (87 boys and 86 girls) and 346 transcriptions.

### 4 Methods

Since the data consist of speech produced by 4- to 7-year-old children, the predictive features must depend on the level of syntactic competence of this age group. The “continuity assumption” states that children have a complete system of abstract syntactic representation and have the same set of abstract functional categories accessible to adults (Pinker, 1984). An experimental study with 3- to 8-year-old children showed that their syntactic competence is comparable to that of adults; specifically, children have a productive rule for passive forms which allows them to generalize to previously unheard predicates while following adult-like constraints to avoid over-generalization (Pinker et al., 1987). Recent experiments with syntactic priming showed that children's representations of abstract passive constructions are well-developed as early as age 3 or 4, and young

children are generally able to form passive constructions with both action and non-action verbs (Thatcher et al., 2007). These results suggest that measures of syntactic complexity that are typically used to evaluate adult language could be adapted to child speech, provided that the children are at least 3 or 4 years old.

Here, the complexity of speech is characterized by the length of utterances and by the frequency of dependent and coordinate clauses, with more complex speech consisting of longer utterances and a higher number of subordinate clauses. We segmented the transcriptions into sentences, clauses and *T-units*, which are “minimally terminable units” consisting of a main clause and its dependent clauses (Hunt, 1965; O’Donnell et al., 1967)<sup>1</sup>. Deceptive communication generally has shorter duration and is less detailed than non-deceptive speech (DePaulo et al., 2003), so the length of each type of segment was counted along with frequency features over segments. Here, the frequency of dependent and coordinate clauses per constituent approximate clause-based measures of complexity.

Our approach combines a set of features obtained from a functional dependency grammar (FDG) parser with another (non-overlapping) set of features obtained from a phrase-based grammar parser. We obtained FDG parses of the transcriptions using Connexor’s Machine Syntax parser (Tapanainen and Järvinen, 1997) and extracted the following 5 features:

**ARI** *Automated readability index*. Measures word and sentence difficulty,  $4.71 \frac{c}{w} + 0.5 \frac{w}{s} - 21.43$ , where  $c$  is the number of characters,  $w$  is the number of words, and  $s$  is the number of sentences (Smith and Senter, 1967).

**ASL** *Average sentence length*. The number of words over the number of sentences.

**COM** *Sentence complexity*. The ratio of sentences with  $\geq 2$  finite predicators to those with  $\leq 1$  finite predicator (Stajner and Mitkov, 2012).

**PAS** *Passivity*. The ratio of non-finite main predicators in a passive construction (@-

FMAINV %VP) to the total number of finite (@+FMAINV %VA) and non-finite (@-FMAINV %VA and @-FMAINV %VP) main predicators, including active constructions.

**MCU** Mean number of clauses per utterance.

Additionally, we searched for specific syntactic patterns in phrase-based parses of the data. We used the Stanford probabilistic natural language parser (Klein and Manning, 2003) for constructing these parse trees, the Stanford Tregex utility (Levy and Andrew, 2006) for searching the constructed parse trees, and a tool provided by Lu (2011) which extracts a set of 14 clause-based features in relation to sentence, clause and T-unit constituents.

#### 4.1 Feature analysis

Analysis of variance (ANOVA) was performed on the set of 17 features, shown in Table 1. A one-factor ANOVA across the *truth* and *lie* groups showed three significant feature variations: average sentence length (ASL), sentence complexity (COM), and mean clauses per utterance (MCU). Dependencies between some feature pairs that are positively correlated are shown in Figure 1.

As expected, the number of clauses (MCU) is dependent on sentence length (ASL) ( $r(344) = .92, p < .001$ ). Also, the number of T-units is dependent on the number of clauses: CN/C is correlated with CN/T ( $r(344) = .89, p < .001$ ), CP/C is correlated with CP/T ( $r(344) = .85, p < .001$ ), and DC/C is correlated with DC/T ( $r(344) = .92, p < .001$ ). Other features are completely uncorrelated. For example, the number of passive constructions is independent of sentence length ( $r(344) = -.0020, p > .05$ ), the number of complex nominals per clause is independent of clause length ( $r(344) = .076, p > .05$ ), and the density of dependent clauses is independent of the density of coordinate phrases ( $r(344) = -.027, p > .05$ ).

## 5 Results

We evaluate five classifiers: logistic regression (**LR**), a multilayer perceptron (**MLP**), naïve Bayes (**NB**), a random forest (**RF**), and a support vector machine (**SVM**). Here, naïve Bayes, which assumes conditional independence of the features, and logistic regression, which has a linear decision boundary, are baselines. The MLP includes a variable number of layers of hidden units, which

<sup>1</sup>T-units include single clauses, two or more phrases in apposition, or clause fragments. Generally, coordinate clauses are split into separate T-units, as are clauses interrupted by discourse boundary markers.

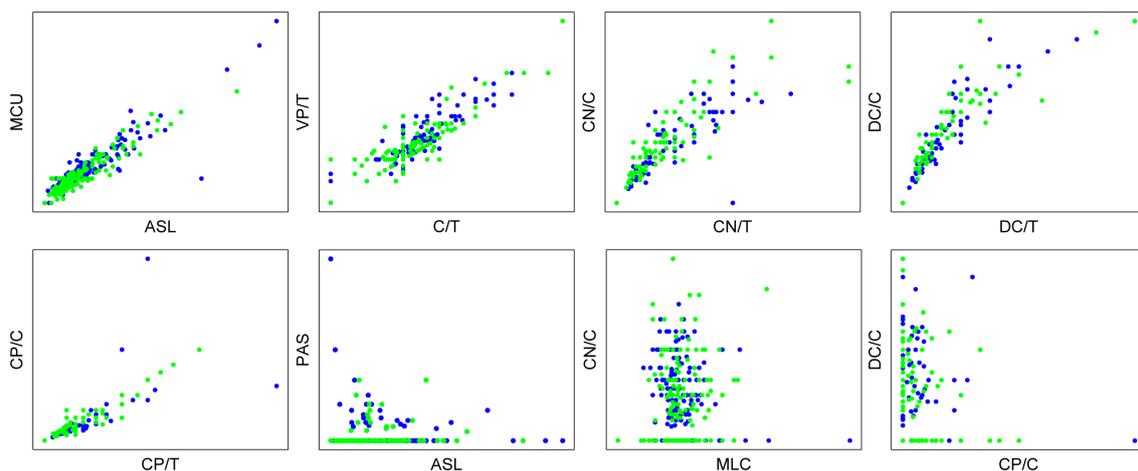


Figure 1: Independent and dependent feature pairs; data points are labeled as *truth* (blue) and *lie* (green).

Feature	$F_{1,344}$	$d$
Automated Readability Index ( <b>ARI</b> )	0.187	0.047
Average Sentence Length ( <b>ASL</b> )	3.870	<b>0.213</b>
Sentence Complexity ( <b>COM</b> )	10.93	<b>0.357</b>
Passive Sentences ( <b>PAS</b> )	1.468	0.131
Mean Clauses per Utterance ( <b>MCU</b> )	6.703	<b>0.280</b>
Mean Length of T-Unit ( <b>MLT</b> )	2.286	0.163
Mean Length of Clause ( <b>MLC</b> )	0.044	-0.023
Verb Phrases per T-Unit ( <b>VP/T</b> )	3.391	0.199
Clauses per T-Unit ( <b>C/T</b> )	2.345	0.166
Dependent Clauses per Clause ( <b>DC/C</b> )	1.207	0.119
Dependent Clauses per T-Unit ( <b>DC/T</b> )	1.221	0.119
T-Units per Sentence ( <b>T/S</b> )	3.692	0.208
Complex T-Unit Ratio ( <b>CT/T</b> )	2.103	0.157
Coordinate Phrases per T-Unit ( <b>CP/T</b> )	0.463	-0.074
Coordinate Phrases per Clause ( <b>CP/C</b> )	0.618	-0.085
Complex Nominals per T-Unit ( <b>CN/T</b> )	0.722	0.092
Complex Nominals per Clause ( <b>CN/C</b> )	0.087	0.032

Table 1: One-factor ANOVA ( $F$  statistics and Cohen’s  $d$ -values,  $\alpha = 0.05$ ) on all features across *truth* and *lie* groups. Statistically significant results are in bold.

apply non-linear activation functions on a linear combination of inputs. The SVM is a parametric binary classifier that provides highly non-linear decision boundaries given particular kernels. The random forest is an ensemble classifier that returns the mode of the class predictions of several decision trees.

### 5.1 Binary classification across all data

The five classifiers were evaluated on the entire pooled data set with 10-fold cross validation. Table 2 lists the parameters varied for each classifier, and Table 3 shows the cross-validation accuracy for the classifiers with the best parameter settings. The naïve Bayes classifier performs poorly, as could be expected given the assumption of conditional feature independence. The SVM classifier

performs best, with 59.5% cross-validation accuracy, which is a statistically significant improvement over the baselines of LR ( $t(4) = 22.25, p < .0001$ ), and NB ( $t(4) = 16.19, p < .0001$ ).

	Parameter	Values
LR	$R$ Ridge value	$10^{-10}$ to $10^{-2}$
	$L$ Learning rate	0.0003 to 0.3
MLP	$M$ Momentum	0 to 0.5
	$H$ Number of hidden layers	1 to 5
	$K$ Use kernel estimator	true, false
RF	$I$ Number of trees	1 to 20
	$K$ Maximum depth	unlimited, 1 to 10
SVM	$K$ Kernel	Linear, RBF, Polynomial
	$E$ Polynomial Exponent	2 to 5
	$G$ RBF Gamma	0.001 to 0.1
	$C$ Complexity constant	0.1 to 10

Table 2: Empirical parameter settings for each classifier

### 5.2 Binary classification by age group

Significant variation in syntactic complexity is expected across ages. To account for such variation, we segmented the dataset in four groups: 44 tran-

	Accuracy	Parameters
LR	0.5347	$R = 10^{-10}$
MLP	0.5838	$L = 0.003, M = 0.4$
NB	0.5173	$K = \text{false}$
RF	0.5809	$I = 10, K = 6$
SVM	0.5954	Polynomial, $E = 3, C = 1$

Table 3: Cross-validation accuracy of binary classification performed on entire dataset of 346 transcriptions.

criptions of 4-year-olds, 120 of 5-year-olds, 94 of 6-year-olds, and 88 of 7-year-olds. By comparison, Vrij et al. (2004) used data from only 35 children in their study of 5- and 6-year-olds. Classification of truthfulness was performed separately for each age, as shown in Table 4. In comparison with classification accuracy on pooled data, a paired  $t$ -test shows statistically significant improvement across all age groups using RF,  $t(3) = 10.37, p < .005$ .

	Age (years)			
	4	5	6	7
LR	0.6136	0.5333	0.5957*	0.4886
MLP	0.6136 <sup>†</sup>	0.5583	0.6170 <sup>†</sup>	0.5909*
NB	0.6136*	0.5250	0.5426	0.5682
RF	0.6364 <sup>†</sup>	<b>0.6333*</b>	<b>0.6383<sup>†</sup></b>	<b>0.6591<sup>†</sup></b>
SVM	<b>0.6591</b>	0.5583	0.6064	0.6250*

Table 4: Cross-validation accuracy of binary classification partitioned by age. The best classifier at each age is shown in bold. The classifiers showing statistically significant incremental improvement are marked: \* $p < .05$ , <sup>†</sup> $p < .001$  (paired  $t$ -test, d.f. 4)

### 5.3 Binary classification by age group, on verbose transcriptions

The length of speech, in number of words, varies widely ( $min = 1, max = 167, \mu = 36.83, \sigma = 28.34$ ) as a result of the unregulated nature of the interview interaction. To test the effect of verbosity, we segment the data by child age and select only the transcriptions with above-average word counts (i.e.,  $\geq 37$  words), resulting in four groups: 12 transcriptions of 4-year-olds, 48 of 5-year-olds, 39 of 6-year-olds, and 37 of 7-year-olds. This mimics the scenario in which some mini-

mum threshold is placed on the length of a child’s speech. In this verbose case, 63.3% of transcripts are labeled truth across age groups (using the same definition of truth as in section 3), with no substantial variation between ages; in the non-verbose case, 53.2% are marked truth. Fisher’s exact test on this contingency table reveals no significant difference between these distributions ( $p = 0.50$ ). Classification results are shown in Table 5. The size of the training set for the youngest age category is low compared to the other age groups, which may reduce the reliability of the higher accuracy achieved in that group. The other three age groups show a growing trend, which is consistent with expectations — older children exhibit greater syntactic complexity in speech, allowing greater variability of feature values across truth and deception. Here, both SVM and RF achieve 83.8% cross-validation accuracy in identifying deception in the speech of 7-year-old subjects.

	4	5	6	7
LR	0.7500 <sup>†</sup>	0.5417	0.6667 <sup>†</sup>	0.7297 <sup>†</sup>
MLP	0.8333 <sup>†</sup>	<b>0.6250<sup>†</sup></b>	0.6154	0.7838 <sup>†</sup>
NB	0.6667 <sup>†</sup>	0.4583	0.4103	0.7297*
RF	0.8333 <sup>†</sup>	0.5625	<b>0.7179<sup>†</sup></b>	<b>0.8378<sup>†</sup></b>
SVM	<b>0.9167*</b>	<b>0.6250<sup>†</sup></b>	0.6154*	<b>0.8378<sup>†</sup></b>

Table 5: Cross-validation accuracy of binary classification performed on transcriptions with above average word count (136 transcriptions), by age group. Rows represent classifiers, columns represent ages. The best classifier for each age is in bold. The classifiers showing statistically significant incremental improvement are marked: \* $p < .05$ , <sup>†</sup> $p < .001$  (paired  $t$ -test, d.f. 4)

### 5.4 Threshold variation

To study the effect of the threshold between the *truth* and *lie* classes, we vary the value of the threshold,  $\tau$ , from 1 to 8, requiring the admission of at least  $\tau$  truthful details (out of 8 possible details) in order to label a transcription as *truth*. The effect of  $\tau$  on classification accuracy over the entire pooled dataset for each of the 5 classifiers is shown in Figure 2. A one-factor ANOVA with  $\tau$  as the independent variable with 8 levels, and cross-validation accuracy as the dependent variable, confirms that the effect of the threshold is statistically significant ( $F_{7,40} = 220.69, p < .0001$ ) with  $\tau = 4$  being the most conservative setting.

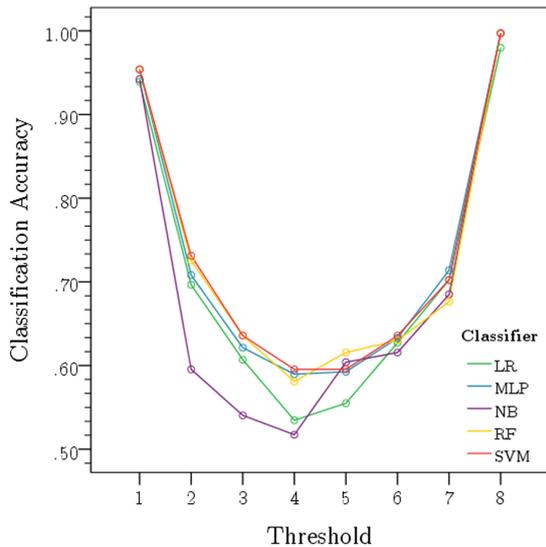


Figure 2: Effect of threshold and classifier choice on cross-validation accuracy. Threshold  $\tau = 0$  is not present, since all data would be labeled *truth*.

### 5.5 Linguistic Inquiry and Word Count

The Linguistic Inquiry and Word Count (LIWC) tool for generating features based on word category frequencies has been used in deception detection with adults, specifically: first-person singular pronouns (FP), exclusive words (EW), negative emotion words (NW), and motion verbs (MV) (Newman et al., 2003). We compare the performance of classifiers trained with our 17 syntactic features to those of classifiers trained with those LIWC-based features on the same data. To evaluate the four LIWC categories, we use the 86 words of the Pennebaker model (Little and Skillicorn, 2008; Vartapetian and Gillam, 2012). The performance of the classifiers trained with LIWC features is shown in Table 6.

The set of 17 syntactic features proposed here result in significantly higher accuracies across classifiers and experiments ( $\mu = 0.63$ ,  $\sigma = 0.10$ ) than with the LIWC features used in previous work ( $\mu = 0.58$ ,  $\sigma = 0.09$ ), as shown in Figure 3 ( $t(53) = -0.0691$ ,  $p < .0001$ ).

## 6 Discussion and future work

This paper evaluates automatic estimation of truthfulness in the utterances of children using a novel set of lexical-syntactic features across five types of classifiers. While previous studies have favored word category frequencies extracted with LIWC (Newman et al., 2003; Little and Skillicorn, 2008; Hauch et al., 2012; Vartapetian and Gillam,

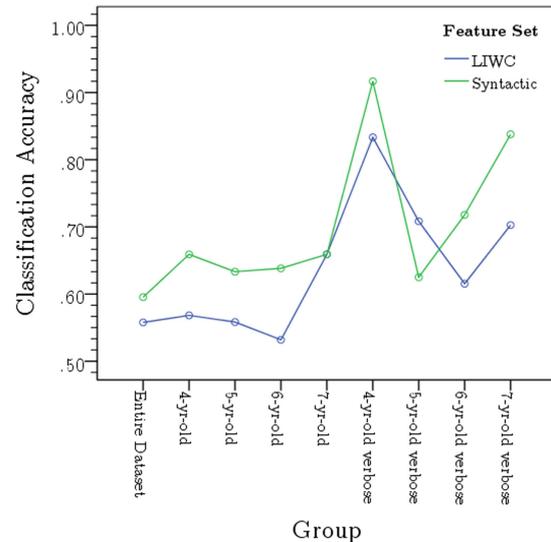


Figure 3: Effect of feature set choice on cross-validation accuracy.

2012; Almela et al., 2012; Fornaciari and Poesio, 2012), our results suggest that the set of syntactic features presented here perform significantly better than the LIWC feature set on our data, and across seven out of the eight experiments based on age groups and verbosity of transcriptions.

Statistical analyses showed that the average sentence length (ASL), the Stajner-Mitkov measure of sentence complexity (COM), and the mean number of clauses per utterance (MCU) are the features most predictive of truth and deception (see section 4.1). Further preliminary experiments are exploring two methods of feature selection, namely forward selection and minimum-Redundancy-Maximum-Relevance (mRMR). In forward selection, features are greedily added one-at-a-time (given an initially empty feature set) until the cross-validation error stops decreasing with the addition of new features (Deng, 1998). This results in a set of only two features: sentence complexity (COM) and T-units per sentence (T/S). Features are selected in mRMR by minimizing redundancy (i.e., the average mutual information between features) and maximizing the relevance (i.e., the mutual information between the given features and the class) (Peng et al., 2005). This approach selects five features: verb phrases per T-unit (VP/T), passive sentences (PAS), coordinate phrases per clause (CP/C), sentence complexity (COM), and complex nominals per clause (CN/C). These results confirm the predictive strength of sentence complexity. Further, preliminary classi-

Group	Accuracy	Best Classifier	Parameters
Entire dataset	0.5578	RF	$I = 20, K = \text{unlimited}$
4-yr-olds	0.5682	MLP	$L = 0.005, M = 0.3, H = 1$
5-yr-olds	0.5583	RF	$I = 5, K = \text{unlimited}$
6-yr-olds	0.5319	MLP	$L = 0.005, M = 0.3, H = 1$
7-yr-olds	0.6591	RF	$I = 5, K = \text{unlimited}$
4-yr-olds, verbose	0.8333	SVM	PolyKernel, $E = 4, C = 10$
5-yr-olds, verbose	0.7083	SVM	NormalizedPolyKernel, $E = 1, C = 10$
6-yr-olds, verbose	0.6154	MLP	$L = 0.09, M = 0.2, H = 1$
7-yr-olds, verbose	0.7027	MLP	$L = 0.01, M = 0.5, H = 3$

Table 6: Best 10-fold cross-validation accuracies achieved on various subsets of the data, using the LIWC-based feature set.

fication results across all classifiers suggest that accuracies are significantly higher given forward selection ( $\mu = 0.58, \sigma = 0.02$ ) relative to the original feature set ( $\mu = 0.56, \sigma = 0.03$ );  $t(5) = -2.28, p < .05$  while the results given the mRMR features are not significantly different.

Generalized cross-validation accuracy increases significantly given partitioned age groups, which suggests that the importance of features may be moderated by age. A further incremental increase is achieved by considering only transcriptions above a minimum length. O’Donnell et al. (1967) examined syntactic complexity in the speech and writing of children aged 8 to 12, and found that speech complexity increases with age. This phenomenon appears to be manifested in the current study by the extent to which classification increases generally across the 5-, 6-, and 7-year-old groups, as shown in Table 5. Future examination of the effect of age on feature saliency may yield more appropriate age-dependent features.

While past research has used logistic regression as a binary classifier (Newman et al., 2003), our experiments show that the best-performing classifiers allow for highly non-linear class boundaries; SVM and RF models achieve between 62.5% and 91.7% accuracy across age groups — a significant improvement over the baselines of LR and NB, as well as over previous results. Moreover, since the performance of human judges in identifying deception is not significantly better than chance (Lewis et al., 1989; Newman et al., 2003), these results show promise in the use of automatic detection methods.

Partially truthful transcriptions were scored using a gradient of 0 to 8 truthful details, and a threshold  $\tau$  was used to perform binary classifica-

tion. Extreme values of  $\tau$  lead to poor F-scores despite high accuracy, since the class distribution of transcriptions is very skewed towards either class. Future work can explore the effect of threshold variation given sufficient data with even class distributions for each threshold setting. When such data is unavailable, experiments can make use of the most conservative setting ( $\tau = 4$ , or an equivalent mid-way setting) for analysis of real-life utterances containing partial truths.

Future work should consider measures of confidence for each classification, where possible, so that more ambiguous classifications are not treated on-par with more certain ones. For instance, confidence can be approximated in MLPs by the entropy across continuous-valued output nodes, and in RFs by the number of component decision trees that agree on a classification. Although acoustic data were not provided with this data set (Lyon and Dorado, 2008) (and, in practice, cannot be assured), future work should also examine the differences in the acoustics of children across truth conditions.

## Acknowledgments

The authors thank Kang Lee (Ontario Institute for Studies in Education, University of Toronto) and Angela Evans (Brock University) for sharing both this data set and their insight.

## Appendix

The following is an example of evasive deceptive speech from a 6-year-old after no truth induction (i.e., the *control* condition in which the interviewer merely states that he needs to ask more questions):

... Yeah yeah ok, I'm a tell you. We played that same game and I won and he won. I'm going to be in trouble if I tell you. It a secret. It's a secret 'cuz we're friends. ...

**Transcription excerpt labeled as *truth* by a threshold of  $\tau = 1$ :** 7-year-old child's response (*play, no coach* condition), in which the child does not explicitly deny playing with the toy house, and admits to looking at it but does not confess to any of the other six actions:

...I was playing, I was hiding the coin and I was trying to find the house... trying to see who was in there...

**Transcription excerpt labeled as *truth* by a threshold of  $\tau = 4$ :** 7-year-old child's response (*play, no coach* condition), in which the child does not explicitly deny playing, and admits to three actions:

...me and him was playing with it... we were just spinning it around and got the toys out...

## References

- Angela Almela, Rafael Valencia-Garcia, and Pascual Cantos. 2012. Seeing through deception: A computational approach to deceit detection in written communication. *Proceedings of the EACL 2012 Workshop on Computational Approaches to Deception Detection*, April 23-27, 2012, Avignon, France, 15-22.
- Ethem Alpaydin. 2010. Introduction to Machine Learning. Cambridge, MA: MIT Press.
- Kan Deng. 1998. OMEGA: On-line memory-based general purpose system classifier. Doctoral thesis, School of Computer Science, Carnegie Mellon University
- Bella M. DePaulo, James J. Lindsay, Brian E. Malone, Laura Muhlenbruck, Kelly Charlton, and Harris Cooper. 2003. Cues to deception. *Psychological Bulletin*, 129(1):74-118.
- Song Feng, Ritwik Banerjee and Yejin Choi. 2012. Syntactic stylometry for deception detection. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, July 8-14, 2012, Jeju, Republic of Korea, 171-175.
- Tommaso Fornaciari and Massimo Poesio. 2012. On the use of homogeneous sets of subjects in deceptive language analysis. *Proceedings of the EACL 2012 Workshop on Computational Approaches to Deception Detection*, April 23-27, 2012, Avignon, France, 39-47.
- Martin Graciarena, Elizabeth Shriberg, Andreas Stolcke, Frank Enos, Julia Hirschberg, Sachin Kajarekar. 2006. Combining prosodic lexical and cepstral systems for deceptive speech detection. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I-1033-I-1036.
- Valerie Hauch, Jaume Masip, Iris Blandon-Gitlin, and Siegfried L. Sporer. 2012. Linguistic cues to deception assessed by computer programs: A meta-analysis. *Proceedings of the EACL Workshop on Computational Approaches to Deception Detection*, pages 1-4.
- Julia Hirschberg, Stefan Benus, Jason M. Brenier, Frank Enos, Sarah Friedman, Sarah Gilman, Cynthia Girand, Martin Graciarena, Andreas Kathol, Laura Michaelis, Bryan Pellom, Elizabeth Shriberg, and Andreas Stolcke. 2005. Distinguishing deceptive from non-deceptive speech. *Proceedings of Eurospeech 2005*, pages 1833-1836.
- Kellogg W. Hunt. 1965. Grammatical structures written at three grade levels. *NCTE Research Report No. 3*.
- Dan Klein and Christopher Manning. 2003. Accurate unlexicalized parsing. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423-430.
- Xuan Le, Ian Lancashire, Graeme Hirst, and Regina Jokel. 2011. Longitudinal detection of dementia through lexical and syntactic changes in writing: a case study of three British novelists. *Literary and Linguistic Computing*, 26(4):435-461.
- Roger Levy and Galen Andrew. 2006. Tregex and Tsurgeon: tools for querying and manipulating tree data structures. *5th International Conference on Language Resources and Evaluation*.
- Michael Lewis, Catherine Stanger, and Margaret W. Sullivan. 1989. Deception in 3-year-olds. *Developmental Psychology*, 25(3):439-443.
- A. Little and D. B. Skillicorn. 2008. Detecting deception in testimony. *Proceedings of IEEE International Conference of Intelligence and Security Informatics (ISI 2008)*, June 17-20, 2008, Taipei, Taiwan, 13-18.

- Xiaofei Lu. 2011. Automatic analysis of syntactic complexity in second language writing. *International Journal of Corpus Linguistics*, 15(4):474-496.
- Thomas D. Lyon and J. S. Dorado. 2008. Truth induction in young maltreated children: the effects of oath-taking and reassurance on true and false disclosures. *Child Abuse & Neglect*, 32(7):738-748.
- Thomas D. Lyon, Lindsay C. Malloy, Jodi A. Quas, and Victoria A. Talwar. 2008. Coaching, truth induction, and young maltreated children's false allegations and false denials. *Child Development*, 79(4):914-929.
- François Mairesse, Marilyn A. Walker, Matthias R. Mehl, and Roger K. Moore. 2007. Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research*, 30:457-500.
- Rada Mihalcea and Carlo Strapparava. 2009. The lie detector: explorations in the automatic recognition of deceptive language. *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, August 4, 2009, Suntec, Singapore, 309-312.
- Matthew L. Newman, James W. Pennebaker, Diane S. Berry, and Jane M. Richards. 2003. Lying words: predicting deception from linguistic styles. *PSPB*, 29(5):665-675.
- Roy C. O'Donnell, William J. Griffin, and Raymond C. Norris. 1967. A transformational analysis of oral and written grammatical structures in the language of children in grades three, five, and seven. *PSPB*, 29(5):665-675.
- Steven Pinker. 1984. *Language learnability and language development*, Cambridge, MA: Harvard University Press.
- Steven Pinker, David S. Lebeaux, and Loren Ann Frost. 1987. Productivity and constraints in the acquisition of the passive. *Cognition*, 26:195-267.
- Hanchuan Peng, Fuhui Long, and Chris Ding. 2005. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226-1238.
- J. M. Richards and J. J. Gross. 1999. Composure at any cost? The cognitive consequences of emotion suppression. *PSPB*, 25(8):1033-1044.
- J. M. Richards and J. J. Gross. 2000. Emotion regulation and memory: the cognitive costs of keeping one's cool. *Journal of Personality and Social Psychology*, 79:410-424.
- E. A. Smith and R. J. Senter. 1967. Automated readability index. Technical report, Defense Technical Information Center. United States.
- Sanja Stajner and Ruslan Mitkov. 2012. Diachronic changes in text complexity in 20th century English language: an NLP Approach. *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 1577-1584.
- Pasi Tapanainen and Timo Järvinen. 1997. A non-projective dependency parser. *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64-71.
- Katherine Thatcher, Holly Branigan, Janet McLean, and Antonella Sorace. 2007. Children's early acquisition of the passive: evidence from syntactic priming. *Child Language Seminar, University of Reading*.
- Udo Undeutsch. 2008. Courtroom evaluation of eyewitness testimony. *Applied Psychology*, 33(1):51-66.
- Anna Vartapetian and Lee Gillam. 2012. "I don't know where he is not": does deception research yet offer a basis for deception detectives? *Proceedings of the EACL 2012 Workshop on Computational Approaches to Deception Detection*, April 23-27, 2012, Avignon, France, 5-14.
- Aldert Vrij, Lucy Akehurst, Stavroula Soukara, and Ray Bull. 2004. Detecting deceit via analyses of verbal and nonverbal behavior in children and adults. *Human Communication Research*, 30(1):8-41
- Yi Zheng, Qifeng Dai, Qiming Luo, and Enhong Chen. 2010. Hedge classification with syntactic dependency features based on an ensemble classifier. *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, July 15-16, 2010, Uppsala, Sweden, 151-156.

# Sentiment Relevance

**Christian Scheible**

Institute for Natural Language Processing  
University of Stuttgart, Germany  
scheibcn@ims.uni-stuttgart.de

**Hinrich Schütze**

Center for Information  
and Language Processing  
University of Munich, Germany

## Abstract

A number of different notions, including subjectivity, have been proposed for distinguishing parts of documents that convey sentiment from those that do not. We propose a new concept, *sentiment relevance*, to make this distinction and argue that it better reflects the requirements of sentiment analysis systems. We demonstrate experimentally that sentiment relevance and subjectivity are related, but different. Since no large amount of labeled training data for our new notion of sentiment relevance is available, we investigate two semi-supervised methods for creating sentiment relevance classifiers: a distant supervision approach that leverages structured information about the domain of the reviews; and transfer learning on feature representations based on lexical taxonomies that enables knowledge transfer. We show that both methods learn sentiment relevance classifiers that perform well.

## 1 Introduction

It is generally recognized in sentiment analysis that only a subset of the content of a document contributes to the sentiment it conveys. For this reason, some authors distinguish the categories *subjective* and *objective* (Wilson and Wiebe, 2003). Subjective statements refer to the internal state of mind of a person, which cannot be observed. In contrast, objective statements can be verified by observing and checking reality. Some sentiment analysis systems filter out objective language and predict sentiment based on subjective language only because objective statements do not directly reveal sentiment.

Even though the categories subjective/objective are well-established in philosophy, we argue that

they are not optimal for sentiment analysis. We instead introduce the notion of *sentiment relevance* (*S-relevance* or *SR* for short). A sentence or linguistic expression is S-relevant if it contains information about the sentiment the document conveys; it is *S-nonrelevant* (*SNR*) otherwise.

Ideally, we would like to have at our disposal a large annotated training set for our new concept of sentiment relevance. However, such a resource does not yet exist. For this reason, we investigate two semi-supervised approaches to S-relevance classification that do not require S-relevance-labeled data. The first approach is distant supervision (DS). We create an initial labeling based on domain-specific metadata that we extract from a public database and show that this improves performance by 5.8%  $F_1$  compared to a baseline. The second approach is transfer learning (TL) (Thrun, 1996). We show that TL improves  $F_1$  by 12.6% for sentiment relevance classification when we use a feature representation based on lexical taxonomies that supports knowledge transfer.

In our approach, we classify *sentences* as S-(non)relevant because this is the most fine-grained level at which S-relevance manifests itself; at the word or phrase level, S-relevance classification is not possible because of scope and context effects. However, S-relevance is also a discourse phenomenon: authors tend to structure documents into S-relevant passages and S-nonrelevant passages. To impose this discourse constraint, we employ a sequence model. We represent each document as a graph of sentences and apply a minimum cut method.

The rest of the paper is structured as follows. Section 2 introduces the concept of sentiment relevance and relates it to subjectivity. In Section 3, we review previous work related to sentiment relevance. Next, we describe the methods applied in this paper (Section 4) and the features we extract (Section 5). Finally, we turn to the description and

results of our experiments on distant supervision (Section 6) and transfer learning (Section 7). We end with a conclusion in Section 8.

## 2 Sentiment Relevance

Sentiment Relevance is a concept to distinguish content informative for determining the sentiment of a document from uninformative content. This is in contrast to the usual distinction between subjective and objective content. Although there is overlap between the two notions, they are different. Consider the following examples for subjective and objective sentences:

(1) Subjective example: *Bruce Banner, a genetics researcher with a tragic past, suffers a horrible accident.*

(2) Objective example: *The movie won a Golden Globe for best foreign film and an Oscar.*

Sentence (1) is subjective because assessments like *tragic past* and *horrible accident* are subjective to the reader and writer. Sentence (2) is objective since we can check the truth of the statement. However, even though sentence (1) has negative subjective content, it is not S-relevant because it is about the plot of the movie and can appear in a glowingly positive review. Conversely, sentence (2) contributes to the positive opinion expressed by the author. Subjectivity and S-relevance are two distinct concepts that do not imply each other: Generally neutral and objective sentences can be S-relevant while certain subjective content is S-nonrelevant. Below, we first describe the annotation procedure for the sentiment relevance corpus and then demonstrate empirically that subjectivity and S-relevance differ.

### 2.1 Sentiment Relevance Corpus

For our initial experiments, we focus on sentiment relevance classification in the movie domain. To create a sentiment-relevance-annotated corpus, the SR corpus, we randomly selected 125 documents from the movie review data set (Pang et al., 2002).<sup>1</sup> Two annotators annotated the sentences for S-relevance, using the labels SR and SNR. If no decision can be made because a sentence contains both S-relevant and S-nonrelevant linguistic material, it is marked as *uncertain*. We excluded 360 sentences that were labeled *uncertain* from the

<sup>1</sup>We used the texts from the raw HTML files since the processed version does not have capitalization.

evaluation. In total, the SR corpus contains 2759 S-relevant and 728 S-nonrelevant sentences. Figure 1 shows an excerpt from the corpus. The full corpus is available online.<sup>2</sup>

First, we study agreement between human annotators. We had 762 sentences annotated for S-relevance by both annotators with an agreement (Fleiss'  $\kappa$ ) of .69. In addition, we obtained subjectivity annotations for the same data on Amazon Mechanical Turk, obtaining each label through a vote of three, with an agreement of  $\kappa = .61$ . However, the agreement of the subjectivity and relevance labelings after voting, assuming that subjectivity equals relevance, is only at  $\kappa = .48$ . This suggests that there is indeed a measurable difference between subjectivity and relevance. An annotator who we asked to examine the 225 examples where the annotations disagree found that 83.5% of these cases are true differences.

### 2.2 Contrastive Classification Experiment

We will now examine the similarities of S-relevance and an existing subjectivity dataset. Pang and Lee (2004) introduced subjectivity data (henceforth *P&L corpus*) that consists of 5000 highly subjective (*quote*) review snippets from rottentomatoes.com and 5000 objective (*plot*) sentences from IMDb plot descriptions.

We now show that although the P&L selection criteria (quotes, plot) bear resemblance to the definition of S-relevance, the two concepts are different.

We use *quote* as S-relevant and *plot* as S-nonrelevant data in TL. We divide both the SR and P&L corpora into training (50%) and test sets (50%) and train a Maximum Entropy (MaxEnt) classifier (Manning and Klein, 2003) with bag-of-word features. Macro-averaged  $F_1$  for the four possible training-test combinations is shown in Table 1. The results clearly show that the classes defined by the two labeled sets are different. A classifier trained on P&L performs worse by about 8% on SR than a classifier trained on SR (68.5 vs. 76.4). A classifier trained on SR performs worse by more than 20% on P&L than a classifier trained on P&L (67.4 vs. 89.7).

Note that the classes are not balanced in the S-relevance data while they are balanced in the subjectivity data. This can cause a misestimation

<sup>2</sup><http://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/sentimentrelevance/>

O	SNR	Braxton is a gambling addict in deep to Mook (Ellen Burstyn), a local bookie.
S	SNR	Kennesaw is bitter about his marriage to a socialite (Rosanna Arquette), believing his wife to be unfaithful.
S	SR	The plot is twisty and complex, with lots of lengthy flashbacks, and plenty of surprises.
S	SR	However, there are times when it is needlessly complex, and at least one instance the storytelling turns so muddled that the answers to important plot points actually get lost.
S	SR	Take a look at L. A. Confidential, or the film’s more likely inspiration, The Usual Suspects for how a complex plot can properly be handled.

Figure 1: Example data from the SR corpus with subjectivity (S/O) and S-relevance (SR/SNR) annotations

		<i>test</i>	
		P&L	SR
<i>train</i>	P&L	89.7	68.5
	SR	67.4	76.4

Table 1: TL/in-task  $F_1$  for P&L and SR corpora

vocabulary	$f_{\text{P&L}}$	$f_{\text{SNR}}$
{actor, director, story}	0	7.5
{good, bad, great}	11.5	4.8

Table 2: % incorrect sentences containing specific words

of class probabilities and lead to the experienced performance drops. Indeed, if we either balance the S-relevance data or unbalance the subjectivity data, we can significantly increase  $F_1$  to 74.8% and 77.9%, respectively, in the noisy label transfer setting. Note however that this step is difficult in practical applications if the actual label distribution is unknown. Also, in a real practical application the distribution of the data is what it is – it cannot be adjusted to the training set. We will show in Section 7 that using an unsupervised sequence model is superior to artificial manipulation of class-imbalances.

An error analysis for the classifier trained on P&L shows that many sentences misclassified as S-relevant ( $f_{\text{SR}}$ ) contain polar words; for example, *Then, the situation turns bad*. In contrast, sentences misclassified as S-nonrelevant ( $f_{\text{SNR}}$ ) contain named entities or plot and movie business vocabulary; for example, *Tim Roth delivers the most impressive acting job by getting the body language right*.

The word count statistics in Table 2 show this for three polar words and for three plot/movie business words. The P&L-trained classifier seems to have a strong bias to classify sentences with po-

lar words as S-relevant even if they are not, perhaps because most training instances for the category *quote* are highly subjective, so that there is insufficient representation of less emphatic S-relevant sentences. These snippets rarely contain plot/movie-business words, so that the P&L-trained classifier assigns almost all sentences with such words to the category S-nonrelevant.

### 3 Related Work

Many publications have addressed subjectivity in sentiment analysis. Two important papers that are based on the original philosophical definition of the term (internal state of mind vs. external reality) are (Wilson and Wiebe, 2003) and (Riloff and Wiebe, 2003). As we argue above, if the goal is to identify parts of a document that are useful/non-useful for sentiment analysis, then S-relevance is a better notion to use.

Researchers have implicitly deviated from the philosophical definition because they were primarily interested in satisfying the needs of a particular task. For example, Pang and Lee (2004) use a minimum cut graph model for review summarization. Because they do not directly evaluate the results of subjectivity classification, it is not clear to what extent their method is able to identify subjectivity correctly.

In general, it is not possible to know what the underlying concepts of a statistical classification are if no detailed annotation guidelines exist and no direct evaluation of manually labeled data is performed.

Our work is most closely related to (Taboada et al., 2009) who define a fine-grained classification that is similar to sentiment relevance on the highest level. However, unlike our study, they fail to experimentally compare their classification scheme to prior work in their experiments and

to show that this scheme is different. In addition, they work on the paragraph level. However, paragraphs often contain a mix of S-relevant and S-nonrelevant sentences. We use the minimum cut method and are therefore able to incorporate discourse-level constraints in a more flexible fashion, giving preference to “relevance-uniform” paragraphs without mandating them.

Täckström and McDonald (2011) develop a fine-grained annotation scheme that includes S-nonrelevance as one of five categories. However, they do not use the category S-nonrelevance directly in their experiments and do not evaluate classification accuracy for it. We do not use their data set as it would cause domain mismatch between the product reviews they use and the available movie review subjectivity data (Pang and Lee, 2004) in the TL approach. Changing both the domain (movies to products) and the task (subjectivity to S-relevance) would give rise to interactions that we would like to avoid in our study.

The notion of annotator rationales (Zaidan et al., 2007) has some overlap with our notion of sentiment relevance. Yessenalina et al. (2010) use rationales in a multi-level model to integrate sentence-level information into a document classifier. Neither paper presents a direct gold standard evaluation of the accuracy of rationale detection.

In summary, no direct evaluation of sentiment relevance has been performed previously. One contribution in this paper is that we provide a single-domain gold standard for sentiment relevance, created based on clear annotation guidelines, and use it for direct evaluation.

Sentiment relevance is also related to review mining (e.g., (Ding et al., 2008)) and sentiment retrieval techniques (e.g., (Eguchi and Lavrenko, 2006)) in that they aim to find phrases, sentences or snippets that are relevant for sentiment, either with respect to certain features or with a focus on high-precision retrieval (cf. (Liu, 2010)). However, finding a few S-relevant items with high precision is much easier than the task we address: exhaustive classification of all sentences.

Another contribution is that we show that generalization based on semantic classes improves S-relevance classification. While previous work has shown the utility of other types of feature generalization for sentiment and subjectivity analysis (e.g., syntax and part-of-speech (Riloff and Wiebe, 2003)), semantic classes have so far not been ex-

ploited.

Named-entity features in movie reviews were first used by Zhuang et al. (2006), in the form of feature-opinion pairs (e.g., a positive opinion about the acting). They show that recognizing plot elements (e.g., script) and classes of people (e.g., actor) benefits review summarization. We follow their approach by using IMDb to define named entity features. We extend their work by introducing methods for labeling partial uses of names and pronominal references. We address a different problem (S-relevance vs. opinions) and use different methods (graph-based and statistical vs. rule-based).

Täckström and McDonald (2011) also solve a similar sequence problem by applying a distantly supervised classifier with an unsupervised hidden sequence component. Their setup differs from ours as our focus lies on pattern-based distant supervision instead of distant supervision using documents for sentence classification.

Transfer learning has been applied previously in sentiment analysis (Tan and Cheng, 2009), targeting polarity detection.

## 4 Methods

Due to the sequential properties of S-relevance (cf. Taboada et al. (2009)), we impose the discourse constraint that an S-relevant (resp. S-nonrelevant) sentence tends to follow an S-relevant (resp. S-nonrelevant) sentence. Following Pang and Lee (2004), we use minimum cut (MinCut) to formalize this discourse constraint.

For a document with  $n$  sentences, we create a graph with  $n + 2$  nodes:  $n$  sentence nodes and source and sink nodes. We define source and sink to represent the classes S-relevance and S-nonrelevance, respectively, and refer to them as SR and SNR.

The individual weight  $\text{ind}(s, x)$  between a sentence  $s$  and the source/sink node  $x \in \{\text{SR}, \text{SNR}\}$  is weighted according to some confidence measure for assigning it to the corresponding class. The weight on the edge from the document’s  $i^{\text{th}}$  sentence  $s_i$  to its  $j^{\text{th}}$  sentence  $s_j$  is set to  $\text{assoc}(s_i, s_j) = c/(j - i)^2$  where  $c$  is a parameter (cf. (Pang and Lee, 2004)). The minimum cut is a tradeoff between the confidence of the classification decisions and “discourse coherence”. The discourse constraint often has the effect that high-confidence labels are propagated over the se-

quence. As a result, outliers with low confidence are eliminated and we get a “smoother” label sequence.

To compute minimum cuts, we use the push-relabel maximum flow method (Cherkassky and Goldberg, 1995).<sup>3</sup>

We need to find values for multiple free parameters related to the sequence model. Supervised optimization is impossible as we do not have any labeled data. We therefore resort to a proxy measure, the *run count*. A run is a sequence of sentences with the same label. We set each parameter  $p$  to the value that produces a median run count that is closest to the true median run count (or, in case of a tie, closest to the true mean run count). We assume that the optimal median/mean run count is known. In practice, it can be estimated from a small number of documents. We find the optimal value of  $p$  by grid search.

## 5 Features

Choosing features is crucial in situations where no high-quality training data is available. We are interested in features that are robust and support generalization. We propose two linguistic feature types for S-relevance classification that meet these requirements.

### 5.1 Generalization through Semantic Features

Distant supervision and transfer learning are settings where exact training data is unavailable. We therefore introduce generalization features which are more likely to support knowledge transfer. To generalize over concepts, we use knowledge from taxonomies. A set of generalizations can be induced by making a cut in the taxonomy and defining the concepts there as base classes. For nouns, the taxonomy is WordNet (Miller, 1995) for which CoreLex (Buitelaar, 1998) gives a set of basic types. For verbs, VerbNet (Kipper et al., 2008) already contains base classes.

We add for each verb in VerbNet and for each noun in CoreLex its base class or basic type as an additional feature where words tagged by the mate tagger (Bohnet, 2010) as NN . \* are treated as nouns and words tagged as VB . \* as verbs. For example, the verb *suggest* occurs in the VerbNet base class *say*, so we add a feature VN:say to the fea-

<sup>3</sup>using the HIPR tool ([www.avglab.com/andrew/soft.html](http://www.avglab.com/andrew/soft.html))

ture representation. We refer to these feature sets as *CoreLex (CX)* and *VerbNet (VN)* features and to their combination as *semantic features (SEM)*.

### 5.2 Named Entities

As standard named entity recognition (NER) systems do not capture categories that are relevant to the movie domain, we opt for a lexicon-based approach similar to (Zhuang et al., 2006). We use the IMDb movie metadata database<sup>4</sup> from which we extract names for the categories <ACTOR>, <PERSONNEL> (directors, screenwriters, and composers), and <CHARACTER> (movie characters). Many entries are unsuitable for NER, e.g., *dog* is frequently listed as a character. We filter out all words that also appear in lower case in a list of English words extracted from the dict.cc dictionary.<sup>5</sup>

A name  $n$  can be ambiguous between the categories (e.g., *John Williams*). We disambiguate by calculating the maximum likelihood estimate of  $p(c|n) = \frac{f(n,c)}{\sum_{c'} f(n,c')}$  where  $c$  is one of the three categories and  $f(n,c)$  is the number of times  $n$  occurs in the database as a member of category  $c$ . We also calculate these probabilities for all tokens that make up a name. While this can cause false positives, it can help in many cases where the name obviously belongs to a category (e.g., *Skywalker* in *Luke Skywalker* is very likely a character reference). We always interpret a name preceding an actor in parentheses as a character mention, e.g., *Reese Witherspoon* in *Tracy Flick (Reese Witherspoon) is an over-achiever [...]* This way, we can recognize character mentions for which IMDb provides insufficient information.

In addition, we use a set of simple rules to propagate annotations to related terms. If a capitalized word occurs, we check whether it is part of an already recognized named entity. For example, if we encounter *Robin* and we previously encountered *Robin Hood*, we assume that the two entities match. Personal pronouns will match the most recently encountered named entity. This rule has precedence over NER, so if a name matches a labeled entity, we do not attempt to label it through NER.

The aforementioned features are encoded as binary presence indicators for each sentence. This

<sup>4</sup>[www.imdb.com/interfaces/](http://www.imdb.com/interfaces/)

<sup>5</sup>[dict.cc](http://dict.cc)

feature set is referred to as *named entities (NE)*.

### 5.3 Sequential Features

Following previous sequence classification work with Maximum Entropy models (e.g., (Ratnaparkhi, 1996)), we use selected features of adjacent sentences. If a sentence contains a feature  $F$ , we add the feature  $F+1$  to the following sentence. For example, if a  $\langle \text{CHARACTER} \rangle$  feature occurs in a sentence,  $\langle \text{CHARACTER}+1 \rangle$  is added to the following sentence. For S-relevance classification, we perform this operation only for NE features as they are restricted to a few classes and thus will not enlarge the feature space notably. We refer to this feature set as *sequential features (SQ)*.

## 6 Distant Supervision

Since a large labeled resource for sentiment relevance classification is not yet available, we investigate semi-supervised methods for creating sentiment relevance classifiers. In this section, we show how to bootstrap a sentiment relevance classifier by distant supervision (DS).

Even though we do not have sentiment relevance annotations, there are sources of metadata about the movie domain that we can leverage for distant supervision. Specifically, movie databases like IMDb contain both metadata about the plot, in particular the characters of a movie, and metadata about the “creators” who were involved in the production of the movie: actors, writers, directors, and composers. On the one hand, statements about characters usually describe the plot and are not sentiment relevant and on the other hand, statements about the creators tend to be evaluations of their contributions – positive or negative – to the movie. We formulate a classification rule based on this observation: Count occurrences of NE features and label sentences that contain a majority of creators (and tied cases) as SR and sentences that contain a majority of characters as SNR. This simple labeling rule covers 1583 sentences with an  $F_1$  score of 67.2% on the SR corpus. We call these labels inferred from NE metadata *distant supervision (DS) labels*. This is a form of distant supervision in that we use the IMDb database as described in Section 5 to automatically label sentences based on which metadata from the database they contain.

To increase coverage, we train a Maximum Entropy (MaxEnt) classifier (Manning and Klein,

2003) on the labels. The MaxEnt model achieves an  $F_1$  of 61.2% on the SR corpus (Table 3, line 2). As this classifier uses training data that is biased towards a specialized case (sentences containing the named entity types creators and characters), it does not generalize well to other S-relevance problems and thus yields lower performance on the full dataset. This distant supervision setup suffers from two issues. First, the classifier only sees a subset of examples that contain named entities, making generalization to other types of expressions difficult. Second, there is no way to control the quality of the input to the classifier, as we have no confidence measure for our distant supervision labeling rule. We will address these two issues by introducing an intermediate step, the unsupervised sequence model introduced in Section 4.

As described in Section 4, each document is represented as a graph of sentences and weights between sentences and source/sink nodes representing SR/SNR are set to the confidence values obtained from the distantly trained MaxEnt classifier. We then apply MinCut as described in the following paragraphs and select the most confident examples as training material for a new classifier.

### 6.1 MinCut Setup

We follow the general MinCut setup described in Section 4. As explained above, we assume that creators and directors indicate relevance and characters indicate nonrelevance. Accordingly, we define  $n_{\text{SR}}$  to be the number of  $\langle \text{ACTOR} \rangle$  and  $\langle \text{PERSONNEL} \rangle$  features occurring in a sentence, and  $n_{\text{SNR}}$  the number of  $\langle \text{CHARACTER} \rangle$  features. We then set the individual weight between a sentence and the source/sink nodes to  $\text{ind}(s, x) = n_x$  where  $x \in \{\text{SR}, \text{SNR}\}$ . The MinCut parameter  $c$  is set to 1; we wish to give the association scores high weights as there might be long spans that have individual weights with zero values.

### 6.2 Confidence-based Data Selection

We use the output of the base classifier to train supervised models. Since the MinCut model is based on a weak assumption, it will make many false decisions. To eliminate incorrect decisions, we only use documents as training data that were labeled with high confidence. As the confidence measure for a document, we use the maximum flow value  $f$  – the “amount of fluid” flowing through the document. The max-flow min-cut theorem (Ford and Fulkerson, 1956) implies that if the flow value

	Model	Features	$F_{SR}$	$F_{SNR}$	$F_m$
1	Majority BL	–	88.3	0.0	44.2
2	MaxEnt (DSlabels)	NE	79.8	42.6	61.2 <sup>1</sup>
3	DSlabels+MinCut	NE	79.6	48.2	63.9 <sup>12</sup>
4	DS MaxEnt	NE	84.8	46.4	65.6 <sup>12</sup>
5	DS MaxEnt	NE+SEM	85.2	48.0	66.6 <sup>124</sup>
6	DS CRF	NE	83.4	49.5	66.4 <sup>12</sup>
7	DS MaxEnt	NE+SQ	84.8	49.2	67.0 <sup>1234</sup>
8	DS MaxEnt	NE+SQ+SEM	84.5	49.1	66.8 <sup>1234</sup>

Table 3: Classification results:  $F_{SR}$  (S-relevant  $F_1$ ),  $F_{SNR}$  (S-nonrelevant  $F_1$ ), and  $F_m$  (macro-averaged  $F_1$ ). Superscript numbers indicate a significant improvement over the corresponding line.

is low, then the cut was found more quickly and thus can be easier to calculate; this means that the sentence is more likely to have been assigned to the correct segment. Following this assumption, we train MaxEnt and Conditional Random Field (CRF, (McCallum, 2002)) classifiers on the  $k\%$  of documents that have the lowest maximum flow values  $f$ , where  $k$  is a parameter which we optimize using the run count method introduced in Section 4.

### 6.3 Experiments and Results

Table 3 shows S-relevant ( $F_{SR}$ ), S-nonrelevant ( $F_{SNR}$ ) and macro average ( $F_m$ )  $F_1$  values for different setups with this parameter. We compare the following setups: (1) The majority baseline (BL) i.e., choosing the most frequent label (SR). (2) a MaxEnt baseline trained on DS labels without application of MinCut; (3) the base classifier using MinCut (DSlabels+MinCut) as described above.

Conditions 4-8 train supervised classifiers based on the labels from DSlabels+MinCut: (4) MaxEnt with named entities (NE); (5) MaxEnt with NE and semantic (SEM) features; (6) CRF with NE; (7) MaxEnt with NE and sequential (SQ) features; (8) MaxEnt with NE, SQ, and SEM.

We test statistical significance using the approximate randomization test (Noreen, 1989) on documents with 10,000 iterations at  $p < .05$ . We achieve classification results above baseline using the MinCut base classifier (line 3) and a considerable improvement through distant supervision. We found that all classifiers using DS labels and MinCut are significantly better than MaxEnt trained on purely rule-based DS labels (line 2). Also, the MaxEnt models using SQ features (lines 7,8) are significantly better than the MinCut base classifier (line 3). For comparison to a chain-based se-

quence model, we train a CRF (line 6); however, the improvement over MaxEnt (line 4) is not significant.

We found that both semantic (lines 5,8) and sequential (lines 7,8) features help to improve the classifier. The best model (line 7) performs better than MinCut (3) by 3.1% and better than training on purely rule-generated DS labels (line 2) by 5.8%. However, we did not find a cumulative effect (line 8) of the two feature sets.

Generally, the quality of NER is crucial in this task. While IMDb is in general a thoroughly compiled database, it is not perfect. For example, all main characters in *Groundhog Day* are listed with their first name only even though the full names are given in the movie. Also, some entries are intentionally incomplete to avoid spoiling the plot. The data also contains ambiguities between characters and titles (e.g., *Forrest Gump*) that are impossible to resolve with our maximum likelihood method. In some types of movies, e.g., documentaries, the distinction between characters and actors makes little sense. Furthermore, ambiguities like occurrences of common names such as *John* are impossible to resolve if there is no earlier full referring expression (e.g., *John Williams*).

Feature analysis for the best model using DS labels (7) shows that NE features are dominant. This correlation is not surprising as the seed labels were induced based on NE features. Interestingly, some subjective features, e.g., *horrible* have high weights for S-nonrelevance, as they are associated with non-relevant content such as plot descriptions.

To summarize, the results of our experiments using distant supervision show that a sentiment relevance classifier can be trained successfully by labeling data with a few simple feature rules, with

MinCut-based input significantly outperforming the baseline. Named entity recognition, accomplished with data extracted from a domain-specific database, plays a significant role in creating an initial labeling.

## 7 Transfer Learning

To address the problem that we do not have enough labeled SR data we now investigate a second semi-supervised method for SR classification, transfer learning (TL). We will use the P&L data (introduced in Section 2.2) for training. This data set has labels that are intended to be subjectivity labels. However, they were automatically created using heuristics and the resulting labels can be either viewed as noisy SR labels or noisy subjectivity labels. Compared to distant supervision, the key advantage of training on P&L is that the training set is much larger, containing around 7 times as much data.

In TL, the key to success is to find a generalized feature representation that supports knowledge transfer. We use a semantic feature generalization method that relies on taxonomies to introduce such features.

We again use MinCut to impose discourse constraints. This time, we first classify the data using a supervised classifier and then use MinCut to smooth the sequences. The baseline (BL) uses a simple bag-of-words representation of sentences for classification which we then extend with semantic features.

### 7.1 MinCut Setup

We again implement the basic MinCut setup from Section 4. We set the individual weight  $\text{ind}(s, x)$  on the edge between sentence  $s$  and class  $x$  to the estimate  $p(x|s)$  returned by the supervised classifier. The parameter  $c$  of the MinCut model is tuned using the run count method described in Section 4.

### 7.2 Experiments and Results

As we would expect, the baseline performance of the supervised classifier on SR is low: 69.9% (Table 4, line 1). MinCut significantly boosts the performance by 7.9% to 77.5% (line 1), a result similar to (Pang and Lee, 2004). Adding semantic features improves supervised classification significantly by 5.7% (75.6% on line 4). When MinCut and both types of semantic features are used together, these improvements are partially cumula-

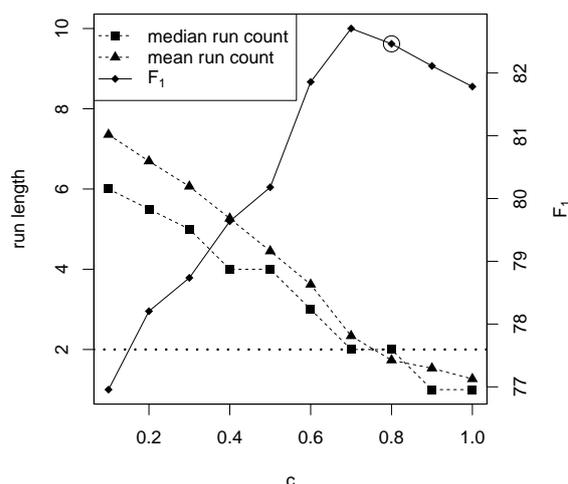


Figure 2:  $F_1$  measure for different values of  $c$ . Horizontal line: optimal median run count. Circle: selected point.

tive: an improvement over the baseline by 12.6% to 82.5% (line 4).

We also experiment with a training set where an artificial class imbalance is introduced, matching the 80:20 imbalance of SR:SNR in the S-relevance corpus. After applying MinCut, we find that while the results for BL with and without imbalances does not differ significantly. However, models using CX and VN features and imbalances are actually significantly inferior to the respective balanced versions. This result suggests that MinCut is more effective at coping with class imbalances than artificial balancing.

MinCut and semantic features are successful for TL because both impose constraints that are more useful in a setup where noise is a major problem. MinCut can exploit test set information without supervision as the MinCut graph is built directly on each test set review. If high-confidence information is “seeded” within a document and then spread to neighbors, mistakes with low confidence are corrected. This way, MinCut also leads to a compensation of different class imbalances.

The results are evidence that semantic features are robust to the differences between subjectivity and S-relevance (cf. Section 2). In the CX+VN model, meaningful feature classes receive high weights, e.g., the *human* class from CoreLex which contains professions that are frequently associated with non-relevant plot descriptions.

To illustrate the run-based parameter optimization criterion, we show  $F_1$  and median/mean run lengths for different values of  $c$  for the best TL

	Model	base classifier			MinCut		
		$F_{SR}$	$F_{SNR}$	$F_m$	$F_{SR}$	$F_{SNR}$	$F_m$
1	BL	81.1	58.6	69.9	87.2	67.8	77.5 <sup>B</sup>
2	CX	82.9	60.1	71.5 <sup>B</sup>	89.0	70.3	79.7 <sup>BM</sup>
3	VN	85.6	62.1	73.9 <sup>B</sup>	91.4	73.6	82.5 <sup>BM</sup>
4	CX+VN	88.3	62.9	75.6 <sup>B</sup>	92.7	72.2	82.5 <sup>BM</sup>

Table 4: Classification results:  $F_{SR}$  (S-relevant  $F_1$ ),  $F_{SNR}$  (S-nonrelevant  $F_1$ ), and  $F_m$  (macro-averaged  $F_1$ ). <sup>B</sup> indicates a significant improvement over the BL base classifier (69.9), <sup>M</sup> over BL MinCut (77.5).

setting (line 4) in Figure 2. Due to differences in the base classifier, the optimum of  $c$  may vary between the experiments. A weaker base classifier may yield a higher weight on the sequence model, resulting in a larger  $c$ . The circled point shows the data point selected through optimization. The optimization criterion does not always correlate perfectly with  $F_1$ . However, we find no statistically significant difference between the selected result and the highest  $F_1$  value.

These experiments demonstrate that S-relevance classification improves considerably through TL if semantic feature generalization and unsupervised sequence classification through MinCut are applied.

## 8 Conclusion

A number of different notions, including subjectivity, have been proposed for distinguishing parts of documents that convey sentiment from those that do not. We introduced *sentiment relevance* to make this distinction and argued that it better reflects the requirements of sentiment analysis systems. Our experiments demonstrated that sentiment relevance and subjectivity are related, but different. To enable other researchers to use this new notion of S-relevance, we have published the annotated S-relevance corpus used in this paper.

Since a large labeled sentiment relevance resource does not yet exist, we investigated semi-supervised approaches to S-relevance classification that do not require S-relevance-labeled data. We showed that a combination of different techniques gives us the best results: semantic generalization features, imposing discourse constraints implemented as the minimum cut graph-theoretic method, automatic “distant” labeling based on a domain-specific metadata database and transfer learning to exploit existing labels for a related classification problem.

In future work, we plan to use sentiment rele-

vance in a downstream task such as review summarization.

## Acknowledgments

This work was funded by the DFG through the Sonderforschungsbereich 732. We thank Charles Jochim, Wiltrud Kessler, and Khalid Al Khatib for many helpful comments and discussions.

## References

- Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China, August. Coling 2010 Organizing Committee.
- P. Buitelaar. 1998. *CoreLex: systematic polysemy and underspecification*. Ph.D. thesis, Brandeis University.
- B. Cherkassky and A. Goldberg. 1995. On implementing push-relabel method for the maximum flow problem. *Integer Programming and Combinatorial Optimization*, pages 157–171.
- X. Ding, B. Liu, and P. S. Yu. 2008. A holistic lexicon-based approach to opinion mining. In *WSDM 2008*, pages 231–240.
- K. Eguchi and V. Lavrenko. 2006. Sentiment retrieval using generative models. In *EMNLP 2006*, pages 345–354.
- L.R. Ford and D.R. Fulkerson. 1956. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399–404.
- K. Kipper, A. Korhonen, N. Ryant, and M. Palmer. 2008. A large-scale classification of English verbs. *Language Resources and Evaluation*, 42(1):21–40.
- B. Liu. 2010. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing*, pages 978–1420085921.
- C. Manning and D. Klein. 2003. Optimization, maxent models, and conditional estimation without magic. In *NAACL-HLT 2003: Tutorials*, page 8.

- A.K. McCallum. 2002. Mallet: A machine learning for language toolkit.
- G.A. Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.
- E.W. Noreen. 1989. *Computer Intensive Methods for Hypothesis Testing: An Introduction*. Wiley.
- B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL 2004*, pages 271–278.
- B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *ACL-EMNLP 2002*, pages 79–86.
- A.M. Popescu and O. Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 339–346. Association for Computational Linguistics.
- A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142.
- E. Riloff and J. Wiebe. 2003. Learning extraction patterns for subjective expressions. In *EMNLP 2003*, pages 105–112.
- M. Taboada, J. Brooke, and M. Stede. 2009. Genre-based paragraph classification for sentiment analysis. In *SIGdial 2009*, pages 62–70.
- O. Täckström and R. McDonald. 2011. Discovering fine-grained sentiment with latent variable structured prediction models. In *ECIR 2011*, pages 368–374.
- S. Tan and X. Cheng. 2009. Improving SCL model for sentiment-transfer learning. In *ACL 2009*, pages 181–184.
- S. Thrun. 1996. Is learning the  $n$ -th thing any easier than learning the first? In *NIPS 1996*, pages 640–646.
- T. Wilson and J. Wiebe. 2003. Annotating opinions in the world press. In *4th SIGdial Workshop on Discourse and Dialogue*, pages 13–22.
- A. Yessenalina, Y. Yue, and C. Cardie. 2010. Multi-level structured models for document-level sentiment classification. In *EMNLP 2010*, pages 1046–1056.
- O. Zaidan, J. Eisner, and C. Piatko. 2007. Using annotator rationales to improve machine learning for text categorization. In *NAACL-HLT 2007*, pages 260–267.
- L. Zhuang, F. Jing, and X. Zhu. 2006. Movie review mining and summarization. In *CIKM 2006*, pages 43–50.

# Predicting and Eliciting Addressee's Emotion in Online Dialogue

Takayuki Hasegawa\*

GREE Inc.

Minato-ku, Tokyo 106-6101, Japan

takayuki.hasegawa@gree.net

Nobuhiro Kaji and Naoki Yoshinaga

Institute of Industrial Science,

the University of Tokyo

Meguro-ku, Tokyo 153-8505, Japan

{kaji, ynaga}@tkl.iis.u-tokyo.ac.jp

Masashi Toyoda

Institute of Industrial Science,

the University of Tokyo

Meguro-ku, Tokyo 153-8505, Japan

toyoda@tkl.iis.u-tokyo.ac.jp

## Abstract

While there have been many attempts to estimate the emotion of an addresser from her/his utterance, few studies have explored how her/his utterance affects the emotion of the addressee. This has motivated us to investigate two novel tasks: predicting the emotion of the addressee and generating a response that elicits a specific emotion in the addressee's mind. We target Japanese Twitter posts as a source of dialogue data and automatically build training data for learning the predictors and generators. The feasibility of our approaches is assessed by using 1099 utterance-response pairs that are built by five human workers.

## 1 Introduction

When we have a conversation, we usually care about the emotion of the person to whom we speak. For example, we try to cheer her/him up if we find out s/he feels down, or we avoid saying things that would trouble her/him.

To date, the modeling of emotion in a dialogue has extensively been studied in NLP as well as related areas (Forbes-Riley and Litman, 2004; Ayadi et al., 2011). However, the past attempts are virtually restricted to estimating the emotion of an addresser<sup>1</sup> from her/his utterance. In contrast, few studies have explored how the emotion of the addressee is affected by the utterance. We consider the insufficiency of such research to be fatal for

<sup>1</sup>This work was conducted while the first author was a graduate student at the University of Tokyo.

<sup>2</sup>We use the terms *addresser/addressee* rather than a speaker/listener, because we target not spoken but online dialogue.



Figure 1: Two example pairs of utterances and responses. Those responses elicit certain emotions, JOY or SADNESS, in the addressee's mind. The addressee in this example refers to the left-hand user, who receives the response.

computers to support human-human communications or to provide a communicative man-machine interface.

With this motivation in mind, the paper investigates two novel tasks: (1) prediction of the addressee's emotion and (2) generation of the response that elicits a prespecified emotion in the addressee's mind.<sup>2</sup> In the prediction task, the system is provided with a dialogue history. For simplicity, we consider, as a history, an utterance and a response to it (Figure 1). Given the history, the system predicts the addressee's emotion that will be caused by the response. For example, the system outputs JOY when the response is *I hope you feel better soon*, while it outputs SADNESS when the response is *Sorry, but you can't join us today*

<sup>2</sup>We adopt Plutchik (1980)'s eight emotional categories in both tasks.

(Figure 1).

In the generation task, on the other hand, the system is provided with an utterance and an emotional category such as JOY or SADNESS, which is referred to as *goal emotion*. Then the system generates the response that elicits the goal emotion in the addressee’s mind. For example, *I hope you feel better soon* is generated as a response to *I have had a high fever for 3 days* when the goal emotion is specified as JOY, while *Sorry, but you can’t join us today* is generated for SADNESS (Figure 1).

Systems that can perform the two tasks not only serve as crucial components of dialogue systems but also have interesting applications of their own. Predicting the emotion of an addressee is useful for filtering flames or infelicitous expressions from online messages (Spertus, 1997). The response generator that is aware of the emotion of an addressee is also useful for text completion in online conversation (Hasselgren et al., 2003; Pang and Ravi, 2012).

This paper explores a data-driven approach to performing the two tasks. With the recent emergence of social media, especially microblogs, the amount of dialogue data available is rapidly increasing. Therefore, we are taking this opportunity to building large-scale training data from microblog posts automatically. This approach allows us to perform the two tasks in a large-scale with little human effort.

We employ standard classifiers for predicting the emotion of an addressee. Our contribution here is to investigate the effectiveness of new features that cannot be used in ordinary emotion recognition, the task of estimating the emotion of a speaker (or writer) from her/his utterance (or writing) (Ayadi et al., 2011; Bandyopadhyay and Okumura, 2011; Balahur et al., 2011; Balahur et al., 2012). We specifically extract features from the addressee’s last utterance (e.g., *I have had a high fever for 3 days* in Figure 1) and explore the effectiveness of using such features. Such information is characteristic of a dialogue situation.

To perform the generation task, we build a statistical response generator by following (Ritter et al., 2011). To improve on the previous study, we investigate a method for controlling the contents of the response for, in our case, eliciting the goal emotion. We achieve this by using a technique inspired by domain adaptation. We learn multiple models, each of which is adapted for eliciting one

specific emotion. Also, we perform model interpolation for addressing data sparseness.

In our experiment, we automatically build training data consisting of over 640 million dialogues from Japanese Twitter posts. Using this data set, we train the classifiers that predict the emotion of an addressee, and the response generators that elicit the goal emotion. We evaluate our methods on the test data that are built by five human workers, and confirm the feasibility of the proposed approaches.

## 2 Emotion-tagged Dialogue Corpus

The key in making a supervised approach to predicting and eliciting addressee’s emotion successful is to obtain large-scale, reliable training data effectually. We thus automatically build a large-scale emotion-tagged dialogue corpus from microblog posts, and use it as the training data in the prediction and generation tasks.

This section describes a method for constructing the emotion-tagged dialogue corpus. We first describe how to extract dialogues from posts in Twitter, a popular microblogging service. We then explain how to automatically annotate utterances in the extracted dialogues with the addressers’ emotions by using emotional expressions as clues.

### 2.1 Mining dialogues from Twitter

We have first crawled utterances (posts) from Twitter by using the Twitter REST API.<sup>3</sup> The crawled data consist of 5.5 billion utterances in Japanese tweeted by 770 thousand users from March 2011 to December 2012. We next cleaned up the crawled utterances by handling Twitter-specific expressions; we replaced all URL strings to ‘URL’, excluded utterances with the symbols that indicate the re-posting (RT) or quoting (QT) of others’ tweets, and erased @user\_name appearing at the head and tail of the utterances, since they are usually added to make a reply. We excluded utterances given by any user whose name included ‘bot.’

We then extracted dialogues from the resulting utterances, assuming that a series of utterances interchangeably made by two users form a dialogue. We here exploited ‘in\_reply\_to\_status\_id’ field of each utterance provided by Twitter REST API to link to the other, if any, utterance to which it replied.

<sup>3</sup><https://dev.twitter.com/docs/api/>

# users	672,937
# dialogues	311,541,839
# unique utterances	1,007,403,858
ave. # dialogues / user	463.0
ave. # utterances / user	1497.0
ave. # utterances / dialogue	3.2

Table 1: Statistics of dialogues extracted from Twitter.

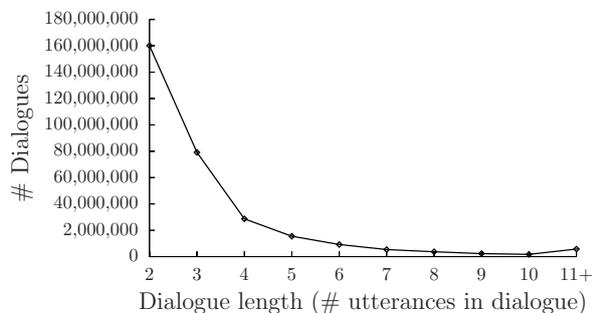


Figure 2: The number of dialogues plotted against the dialogue length.

Utterance	Emotion
A: Would you like to go for dinner with me?	
B: Sorry, I can't. I have a fever of 38 degrees.	
A: Oh dear. I hope you feel better soon.	SURPRISE
B: Thanks. I'm happy to hear you say that.	JOY

Table 2: An illustration of an emotion-tagged dialogue: The first column shows a dialogue (a series of utterances interchangeably made by two users), while the second column shows the addresser's emotion estimated from the utterance.

Table 1 lists the statistics of the extracted dialogues, while Figure 2 plots the number of dialogues plotted against the dialogue length (the number of utterances in dialogue). Most dialogues (98.2%) consist of at most 10 utterances, although the longest dialogue includes 1745 utterances and spans more than six weeks.

## 2.2 Tagging utterances with addressers' emotions

We then automatically labeled utterances in the obtained dialogues with the addressers' emotions by using emotional expressions as clues (Table 2). In this study, we have adopted Plutchik (1980)'s eight emotional categories (ANGER, ANTICIPATION, DISGUST, FEAR, JOY, SADNESS, SURPRISE, and TRUST) as the targets to label, and manually tailored around ten emotional expressions for each emotional category. Table 3 lists examples of the emotional expressions, while the

Emotion	Emotional expressions
ANGER	frustrating, irritating, nonsense
ANTICIPATION	exciting, expecting, looking forward
DISGUST	disgusting, unpleasant, hate
FEAR	afraid, anxious, scary
JOY	glad, happy, delighted
SADNESS	sad, lonely, unhappy
SURPRISE	surprised, oh dear, wow
TRUST	relieved, reliable, solid

Table 3: Example of clue emotional expressions.

Emotion	# utterances	Precision	
		Worker A	Worker B
ANGER	190,555	0.95	0.95
ANTICIPATION	2,548,706	0.99	0.99
DISGUST	475,711	0.93	0.93
FEAR	2,671,222	0.96	0.96
JOY	2,725,235	0.94	0.96
SADNESS	712,273	0.97	0.97
SURPRISE	975,433	0.97	0.97
TRUST	359,482	0.97	0.98

Table 4: Size and precision of utterances labeled with the addressers' emotions.

rest are mostly their spelling variations.<sup>4</sup>

Because precise annotation is critical in the supervised learning scenario, we annotate utterances with the addressers' emotions only when the emotional expressions do not:

1. modify content words.
2. accompany an expression of negation, conditional, imperative, interrogative, concession, or indirect speech in the same sentence.

For example, *I saw a frustrated teacher* is rejected by the first condition, while *I'll be happy if it rains* is rejected by the second condition. The second condition was judged by checking whether the sentence includes trigger expressions such as 'ない (*not/never*)', 'たら (*if-clause*)', '?', 'けど (*(al)though*)', and 'と (*that-clause*)'.

Table 4 lists the size and precision of the utterances labeled with the addressers' emotions. Two human workers measured the precision of the annotation by examining 100 labeled utterances randomly sampled for each emotional category. The inter-rater agreement was  $\kappa = 0.85$ , indicating almost perfect agreement. The precision of the annotation exceeded 0.95 for most of the emotional categories.

<sup>4</sup>Note that the clue emotional expressions are language-specific but can be easily tailored for other languages. Here, Japanese emotional expressions are translated into English to widen the potential readership of the paper.

### 3 Predicting Addressee’s Emotion

This section describes a method for predicting emotion elicited in an addressee when s/he receives a response to her/his utterance. The input to this task is a pair of an utterance and a response to it, e.g., the two utterances in Figure 1, while the output is the addressee’s emotion among the emotional categories of Plutchik (1980) (JOY and SADNESS for the top and bottom dialogues in Figure 1, respectively).

Although a response could elicit multiple emotions in the addressee, in this paper we focus on predicting the most salient emotion elicited in the addressee and cast the prediction as a single-label multi-class classification problem.<sup>5</sup> We then construct a one-versus-the-rest classifier<sup>6</sup> by combining eight binary classifiers, each of which predicts whether the response elicits each emotional category. We use online passive-aggressive algorithm to train the eight binary classifiers.

We exploit the emotion-tagged dialogue corpus constructed in Section 2 to collect training examples for the prediction task. For each emotion-tagged utterance in the corpus, we assume that the tagged emotion is elicited by the (last) response. We thereby extract the pair of utterances preceding the emotion-tagged utterance and the tagged emotion as one training example. Taking the dialogue in Table 2 as an example, we obtain one training example from the first two utterances and SURPRISE as the emotion elicited in user A.

We extract all the  $n$ -grams ( $n \leq 3$ ) in the response to induce (binary)  $n$ -gram features. The extracted  $n$ -grams could indicate a certain action that elicits a specific emotion (e.g., ‘have a fever’ in Table 2), or a style or tone of speaking (e.g., ‘Sorry’). Likewise, we extract word  $n$ -grams from the addressee’s utterance. The extracted  $n$ -grams activate another set of binary  $n$ -gram features.

Because word  $n$ -grams themselves are likely to be sparse, we estimate the addressers’ emotions from their utterances and exploit them to induce emotion features. The addresser’s emotion has been reported to influence the addressee’s emotion

<sup>5</sup>Because microblog posts are short, we expect emotions elicited by a response post not to be very diverse and a multi-class classification to be able to capture the essential crux of the prediction task.

<sup>6</sup>We should note that a one-versus-the-rest classifier can be used in the multi-label classification scenario, just by allowing the classifier to output more than one emotional category (Ghamrawi and McCallum, 2005).

strongly (Kim et al., 2012), while the addressee’s emotion just before receiving a response can be a reference to predict her/his emotion in question after receiving the response.

To induce emotion features, we exploit the rule-based approach used in Section 2.2 to estimate the addresser’s emotion. Since the rule-based approach annotates utterances with emotions only when they contain emotional expressions, we independently train for each emotional category a binary classifier that estimates the addresser’s emotion from her/his utterance and apply it to the unlabeled utterances. The training data for these classifiers are the emotion-tagged utterances obtained in Section 2, while the features are  $n$ -grams ( $n \leq 3$ )<sup>7</sup> in the utterance.

We should emphasize that the features induced from the addressee’s utterance are unique to this task and are hardly available in the related tasks that predicted the emotion of a reader of news articles (Lin and Hsin-Yih, 2008) or personal stories (Socher et al., 2011). We will later confirm the impact of these features on the prediction accuracy in the experiments.

### 4 Eliciting Addressee’s Emotion

This section presents a method for generating a response that elicits the goal emotion, which is one of the emotional categories of Plutchik (1980), in the addressee. In section 4.1, we describe a statistical framework for response generation proposed by (Ritter et al., 2011). In section 4.2, we present how to adapt the model in order to generate a response that elicits the goal emotion in the addressee.

#### 4.1 Statistical response generation

Following (Ritter et al., 2011), we apply the statistical machine translation model for generating a response to a given utterance. In this framework, a response is viewed as a translation of the input utterance. Similar to ordinary machine translation systems, the model is learned from pairs of an utterance and a response by using off-the-shelf tools for machine translation.

We use GIZA++<sup>8</sup> and SRILM<sup>9</sup> for learning translation model and 5-gram language model, re-

<sup>7</sup>We have excluded  $n$ -grams that matched the emotional expressions used in Section 2 to avoid overfitting.

<sup>8</sup><http://code.google.com/p/giza-pp/>

<sup>9</sup><http://www.speech.sri.com/projects/srilm/>

spectively. As post-processing, some phrase pairs are filtered out from the translation table as follows. When GIZA++ is directly applied to dialogue data, it frequently finds paraphrase pairs, learning to parrot back the input (Ritter et al., 2011). To avoid using such pairs for response generation, a phrase pair is removed if one phrase is the substring of the other.

We use Moses decoder<sup>10</sup> to search for the best response to a given utterance. Unlike machine translation, we do not use reordering models, because the positions of phrases are not considered to correlate strongly with the appropriateness of responses (Ritter et al., 2011). In addition, we do not use any discriminative training methods such as MERT for optimizing the feature weights (Och, 2003). They are set as default values provided by Moses (Ritter et al., 2011).

## 4.2 Model adaptation

The above framework allows us to generate appropriate responses to arbitrary input utterances. On top of this framework, we have developed a response generator that elicits a specific emotion.

We use the emotion-tagged dialogue corpus to learn eight translation models and language models, each of which is specialized in generating the response that elicits one of the eight emotions (Plutchik, 1980). Specifically, the models are learned from utterances preceding ones that are tagged with emotional category. As an example, let us examine to learn models for eliciting SURPRISE from the dialogue in Table 2. In this case, the first two utterances are used to learn the translation model, while only the second utterance is used to learn the language model.

However, this simple approach is prone to suffer from the data sparseness problem. Because not all the utterances are tagged with the emotion in emotion-tagged dialogue corpus, only a small fraction of utterances can be used for learning the adapted models.

We perform model interpolation for addressing this problem. In addition to the adapted models described above, we also use a general model, which is learned from the entire corpus. The two models are then merged as the weighted linear interpolation.

Specifically, we use `tmcombine.py` script provided by Moses for the interpolation of trans-

lation models (Sennrich, 2012). For all the four features (i.e., two phrase translation probabilities and two lexical weights) derived from translation model, the weights of the adapted model are equally set as  $\alpha$  ( $0 \leq \alpha \leq 1.0$ ). On the other hand, we use SRILM for the interpolation of language models. The weight of the adapted model is set as  $\beta$  ( $0 \leq \beta \leq 1.0$ ).

The parameters  $\alpha$  and  $\beta$  control the strength of the adapted models. Only adapted models are used when  $\alpha$  (or  $\beta$ ) = 1.0, while the adapted models are not at all used when  $\alpha$  (or  $\beta$ ) = 0. When both  $\alpha$  and  $\beta$  are specified as 0, the model becomes equivalent to the original one described in section 4.1.

## 5 Experiments

### 5.1 Test data

To evaluate the proposed method, we built, as test data, sets of an utterance paired with responses that elicit a certain goal emotion (Table 5). Note that they were used for evaluation in both of the two tasks. Each utterance in the test data has more than one responses that elicit the same goal emotion, because they are used to compute BLEU score (see section 5.3).

The data set was built in the following manner. We first asked five human worker to produce responses to 80 utterances (10 utterances for each goal emotion). Note that the 80 utterances do not have overlap between workers and that the worker produced only one response to each utterance.

To alleviate the burden on the workers, we actually provided each worker with the utterances in the emotion-tagged corpus. Then we asked each worker to select 80 utterances to which s/he thought s/he could easily respond. The selected utterances were removed from the corpus during training.

As a result, we obtained 400 utterance-response pairs (= 80 utterance-response pairs  $\times$  5 workers). For each of those 400 utterances, two additional responses are produced. We did not allow the same worker to produce more than one response to the same utterance. In this way, we obtained 1200 responses for the 400 utterances in total.

Finally, we assessed the data quality to remove responses that were unlikely to elicit the goal emotion. For each utterance-response pair, we asked two workers to judge whether the response elicited the goal emotion. If both workers regarded the

<sup>10</sup><http://www.statmt.org/moses/>

Goal emotion: JOY
U: 16歳になりました, これからもよろしくお願 いします! (I'm turning 16. Hope to get along with you as well as ever!)
R1: 誕生日おめでとうございます! (Happy birthday!)
R2: おめでとう! 今度誕生日プレゼントあげるね. (Congratulations! I'll give you a birthday present.)
R3: おめでとうー!! 幸せな一年を! (Congratulations! I hope you have a happy year!)

Table 5: Example of the test data. English translations are attached in the parenthesis.

Emotion	# utterance pairs
ANGER	119,881
ANTICIPATION	1,416,847
DISGUST	333,972
FEAR	1,662,998
JOY	1,724,198
SADNESS	436,668
SURPRISE	589,790
TRUST	228,974
GENERAL	646,429,405

Table 6: The number of utterance pairs used for training classifiers in emotion prediction and learning the translation models and language models in response generation.

response as inappropriate, it was removed from the data. The resulting test data consist of 1099 utterance-response pairs for 396 utterances.

This data set is submitted as supplementary material to support the reproducibility of our experimental results.

## 5.2 Prediction task

We first report experimental results on predicting the addressee's emotion within a dialogue. Table 6 lists the number of utterance-response pairs used to train eight binary classifiers for individual emotional categories, which form a one-versus-the rest classifier for the prediction task. We used opal<sup>11</sup> as an implementation of online passive-aggressive algorithm to train the individual classifiers.

To investigate the impact of the features that are uniquely available in a dialogue data, we compared classifiers trained with the following two sets of features in terms of precision, recall, and F<sub>1</sub> for each emotional category.

**RESPONSE** The  $n$ -gram and emotion features induced from the response.

<sup>11</sup><http://www.tkl.iis.u-tokyo.ac.jp/~ynaga/opal/>.

Emotion	RESPONSE			RESPONSE/UTTER.		
	PREC	REC	F <sub>1</sub>	PREC	REC	F <sub>1</sub>
ANGER	0.455	0.476	0.465	<b>0.600</b>	<b>0.548</b>	<b>0.573</b>
ANTICIPA.	0.518	0.526	0.522	<b>0.614</b>	<b>0.637</b>	<b>0.625</b>
DISGUST	0.275	<b>0.519</b>	0.359	<b>0.378</b>	0.511	<b>0.435</b>
FEAR	<b>0.484</b>	<b>0.727</b>	<b>0.581</b>	0.459	0.706	0.556
JOY	0.690	0.417	0.519	<b>0.720</b>	<b>0.590</b>	<b>0.649</b>
SADNESS	<b>0.711</b>	0.467	0.564	0.670	<b>0.562</b>	<b>0.611</b>
SURPRISE	0.511	0.348	0.414	<b>0.584</b>	<b>0.437</b>	<b>0.500</b>
TRUST	<b>0.695</b>	0.452	0.548	0.682	<b>0.514</b>	<b>0.586</b>
average	0.542	0.492	0.497	<b>0.588</b>	<b>0.563</b>	<b>0.567</b>

Table 7: Predicting addressee's emotion: Results.

	PREDICTED EMOTION								total
	ANGER	ANTICIPA.	DISGUST	FEAR	JOY	SADNESS	SURPRISE	TRUST	
ANGER	<b>69</b>	0	<u>26</u>	20	0	8	2	1	126
ANTICIPA.	1	<b>86</b>	11	7	<u>13</u>	0	6	11	135
DISGUST	<u>25</u>	1	<b>68</b>	18	2	8	7	4	133
FEAR	3	0	<u>22</u>	<b>101</b>	1	5	9	2	143
JOY	1	<u>28</u>	9	4	<b>85</b>	1	7	9	144
SADNESS	6	3	<u>25</u>	14	5	<b>77</b>	5	2	137
SURPRISE	7	10	9	<u>32</u>	5	7	<b>59</b>	6	135
TRUST	3	12	10	<u>24</u>	7	9	6	<b>75</b>	146
total	115	140	180	220	118	115	101	110	1099

Table 8: Confusion matrix of predicting addressee's emotion, with mostly predicted emotions **bold-faced** and mostly confused emotions underlined for each emotional category.

**RESPONSE/UTTER.** The  $n$ -gram and emotion features induced from the response and the addressee's utterance.

Table 7 lists prediction results. We can see that the features induced from the addressee's utterance significantly improved the prediction performance, F<sub>1</sub>, for emotions other than FEAR. FEAR is elicited instantly by the response, and the features induced from the addressee's utterance thereby confused the classifier.

Table 8 shows a confusion matrix of the classifier using all the features, with mostly predicted emotions bold-faced and mostly confused emotions underlined for each emotional category. We can find some typical confusing pairs of emotions from this matrix. The classifier confuses DISGUST with ANGER and vice versa, while it confuses JOY with ANTICIPATION. These confusions conform to our expectation, since they are actually similar emotions. The classifier was less likely to confuse positive emotions (JOY and ANTICIPATION) with negative emotion (ANGER, DISGUST, FEAR, and SADNESS) vice versa.

<b>Goal emotion:</b> ANGER (predicted as SADNESS)
<b>U:</b> 毎日通話してるなんなの羨ましいわ (You have phone calls every day, I envy you.)
<b>R:</b> 君の方こそ誰からも電話こないから暇で羨ましいよ。 (I envy you have a lot of time 'cause no one calls you.)
<b>Goal emotion:</b> SURPRISE (predicted as FEAR)
<b>U:</b> 黒髪がモテるってマジか。 (Is it true that dark-haired girls are popular with boys?)
<b>R:</b> 80%くらいの男子は黒髪が好きらしい。 (About 80% of boys seem to prefer dark-haired girls.)

Table 9: Examples of utterance-response pairs to which the system predicted wrong emotions.

We have briefly examined the confusions and found the two major types of errors, each of which is exemplified in Table 9. The first (top) one is sarcasm or irony, which has been reported to be difficult to capture by lexical features alone (González-Ibáñez et al., 2011). The other (bottom) one is due to lack of information. In this example, only if the addressee does not know the fact provided by the response, s/he will surprise at it.

### 5.3 Generation task

We next demonstrate the experimental results for eliciting the emotion of the addressee.

We use the utterance pairs summarized in Table 6 to learn the translation models and language models for eliciting each emotional category. We also use the 640 million utterances pairs in the entire emotion-tagged corpus for learning general models. However, for learning the general translation models, we currently use 4 millions of utterance pairs sampled from the 640 millions of pairs due to the computational limitation.

#### Automatic evaluation

We first use BLEU score (Papineni et al., 2002) to perform automatic evaluation (Ritter et al., 2011). In this evaluation, the system is provided with the utterance and the goal emotion in the test data and the generated responses are evaluated through BLEU score. Specifically, we conducted two-fold cross-validation to optimize the weights of our method. We tried  $\alpha$  and  $\beta$  in  $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$  and selected the weights that achieved the best BLEU score. Note that we adopted different values of the weights for different emotional categories.

Table 10 compares BLEU scores of three methods including the proposed one. The first row represents a method that does not perform model adaptation at all. It corresponds to the special case

System	BLEU
NO ADAPTATION	0.64
PROPOSED	1.05
OPTIMAL	1.57

Table 10: Comparison of BLEU scores.

(i.e.,  $\alpha = \beta = 0.0$ ) of the proposed method. The second row represents our method, while the last row represents the result of our method when the weights are set as optimal, i.e., those achieving the best BLEU on the test data. This result can be considered as an upper bound on BLEU score.

The results demonstrate that model adaptation is useful for generating the responses that elicit the goal emotion. We can clearly observe the improvement in the BLEU from 0.64 to 1.05.

On the other hand, there still remains a gap between the last two rows (i.e., proposed and optimal). We think this is partly because the current test data is too small to reliably tune parameters.

#### Human evaluation

We next asked two human workers to manually evaluate the generation results.

In this evaluation, the baseline (no adaptation in Table 10) and proposed method generated a response for each of the 396 utterances in the test data. For the resulting 792 utterance-response pairs, the two workers manually assessed the appropriateness of the response. Each response was judged whether it is grammatical and meaningful. If the response was regarded as so by either of the workers, it was further judged whether it elicits the goal emotion or not. To make the comparison fair, we did not expose to the workers which system generated the response. In addition, the responses generated by the two systems were presented in a random order.

As the result, 147 and 157 responses of the baseline and proposed method were regarded as appropriate, i.e., eliciting the goal emotion, by either of the workers; 74 and 92 responses were regarded as appropriate by both of the workers. These results suggest the effectiveness of the proposed method. Especially, we can confirm that the proposed method can generate responses that elicit addressee’s emotion more clearly. We investigated the agreement between the two workers in this evaluation. We found that the  $\kappa$  coefficient is 0.59, which indicates moderate agreement. This supports the reliability of our evaluation.

<b>Goal emotion:</b>	JOY
<b>Input:</b>	写真コンクールで優秀賞とったーーーー！2月7日表彰式！ (I woooooon the outstanding award at the photo competition! The ceremony is on Feb. 7!)
NO ADAPTATION:	残念ながら1日目だけですのー (Sorry to say, only the first day.)
PROPOSED:	金賞受賞おめでとう!!! おめでとうなのー (Congratulations on winning the gold prize!!! Congrats.)
<b>Goal emotion:</b>	TRUST
<b>Input:</b>	自暴自棄になっているの？ (Do you get desperate?)
NO ADAPTATION:	楽しみ！ (I'm looking forward to it!)
PROPOSED:	きっとまだ大丈夫 (Maybe still OK.)
<b>Goal emotion:</b>	ANTICIPATION
<b>Input:</b>	だよね！なんとかなるよね！ww (Huh! It's gonna be all right! lol) あ、わたしグッズ買わなきゃなのでその時間だけは取ってくれるとうれしい(´▽`) (I gotta buy the goods, so I'll be glad if you can take the time :-))
NO ADAPTATION:	私はグッズ買ってないから不安ですね (Since I've not bought it, I feel worried.)
PROPOSED:	いいですね！私も買いますね!!! (Good! I'll buy it too!!!)

Table 11: Examples of the responses generated by the two systems, NO ADAPTATION and PROPOSED.

## Examples

Table 11 illustrates examples of the responses generated by the no adaptation baseline and proposed method. In the first two examples, the proposed method successfully generates responses that elicit the goal emotions: JOY and TRUST. From these examples, we can consider that the adapted model assigns large probability to phrases such as *congratulations* or *OK*. In the last example, the system also succeeded in eliciting the goal emotion: ANTICIPATION. For this example, we can interpret that the speaker of the response (i.e., the system) feels anticipation, and consequently the emotion of the addressee is affected by the emotion of the speaker (i.e., the system). Interestingly, a similar phenomenon is also observed in real conversation (Kim et al., 2012).

## 6 Related Work

There have been a tremendous amount of studies on predicting the emotion from text or speech data (Ayadi et al., 2011; Bandyopadhyay and Okumura, 2011; Balahur et al., 2011; Balahur et al., 2012). Unlike our prediction task, most of them have exclusively focused on estimating the emotion of a speaker (or writer) from her/his utterance (or writing).

Analogous to our prediction task, Lin and Hsin-Yih (2008) and Socher et al. (2011) investigated predicting the emotion of a reader from the text that s/he reads. Our work differs from them in that we focus on dialogue data, and we exploit features that are not available within their task settings, e.g., the addressee's previous utterance.

Tokuhisa et al. (2008) proposed a method for

extracting pairs of an event (e.g., *It rained suddenly when I went to see the cherry blossoms*) and an emotion elicited by it (e.g., SADNESS) from the Web text. The extracted data are used for emotion classification. A similar technique would be useful for prediction the emotion of an addressee as well.

Response generation has a long research history (Weizenbaum, 1966), although it is only very recently that a fully statistical approach was introduced in this field (Ritter et al., 2011). At this moment, we are unaware of any statistical response generators that model the emotion of the user.

Some researchers have explored generating jokes or humorous text (Dybala et al., 2010; Labtov and Lipson, 2012). Those attempts are similar to our work in that they also aim at eliciting a certain emotion in the addressee. They are, however, restricted to elicit a specific emotion.

The linear interpolation of translation and/or language models is a widely-used technique for adapting machine translation systems to new domains (Sennrich, 2012). However, it has not been touched in the context of response generation.

## 7 Conclusion and Future Work

In this paper, we have explored predicting and eliciting the emotion of an addressee by using a large amount of dialogue data obtained from microblog posts. In the first attempt to model the emotion of an addressee in the field of NLP, we demonstrated that the response of the dialogue partner and the previous utterance of the addressee are useful for predicting the emotion. In the generation task, on the other hand, we showed that the

model adaptation approach successfully generates the responses that elicit the goal emotion.

For future work, we want to use longer dialogue history in both tasks. While we considered only two utterances as a history, a longer history would be helpful. We also plan to personalize the proposed methods, exploiting microblog posts made by users of a certain age, gender, occupation, or even character to perform model adaptation.

## Acknowledgment

This work was supported by the FIRST program of JSPS. The authors thank the anonymous reviewers for their valuable comments. The authors also thank the student annotators for their hard work.

## References

- Moataz El Ayadi, Mohamed S. Kamel, and Fakhri Karay. 2011. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44:572–587.
- Alexandra Balahur, Ester Boldrini, Andres Montoyo, and Patricio Martinez-Barco, editors. 2011. *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*. Association for Computational Linguistics.
- Alexandra Balahur, Andres Montoyo, Patricio Martinez Barco, and Ester Boldrini, editors. 2012. *Proceedings of the 3rd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*. Association for Computational Linguistics.
- Sivaji Bandyopadhyay and Manabu Okumura, editors. 2011. *Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology*. Asian Federation of Natural Language Processing.
- Pawel Dybala, Michal Ptaszynski, Jacek Maciejewski, Mizuki Takahashi, Rafal Rzepka, and Kenji Araki. 2010. Multiagent system for joke generation: Humor and emotions combined in human-agent conversation. *Journal of Ambient Intelligence and Smart Environments*, 2(1):31–48.
- Kate Forbes-Riley and Diane J. Litman. 2004. Predicting emotion in spoken dialogue from multiple knowledge sources. In *Proceedings of NAACL*, pages 201–208.
- Nadia Ghamrawi and Andrew McCallum. 2005. Collective multi-label classification. In *Proceedings of CIKM*, pages 195–200.
- Roberto González-Ibáñez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of ACL*, pages 581–586.
- Jon Hasselgren, Erik Montnemery, Pierre Nugues, and Markus Svensson. 2003. HMS: A predictive text entry method using bigrams. In *Proceedings of EACL Workshop on Language Modeling for Text Entry Methods*, pages 43–50.
- Suin Kim, JinYeong Bak, and Alice Haeyun Oh. 2012. Do you feel what I feel? social aspects of emotions in Twitter conversations. In *Proceedings of ICWSM*, pages 495–498.
- Igor Labtov and Hod Lipson. 2012. Humor as circuits in semantic networks. In *Proceedings of ACL (Short Papers)*, pages 150–155.
- Kevin Lin and Hsin-Hsi Hsin-Yih. 2008. Ranking reader emotions using pairwise loss minimization and emotional distribution regression. In *Proceedings of EMNLP*, pages 136–144.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167.
- Bo Pang and Sujith Ravi. 2012. Revisiting the predictability of language: Response completion in social media. In *Proceedings of EMNLP*, pages 1489–1499.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.
- Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. In *Emotion: Theory, research, and experience: Vol. 1. Theories of emotion*, pages 3–33. New York: Academic.
- Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-driven response generation in social media. In *Proceedings of EMNLP*, pages 583–593.
- Rico Sennrich. 2012. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of EACL*, pages 539–549.
- Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of EMNLP*, pages 151–161.
- Ellen Spertus. 1997. Smokey: Automatic recognition of hostile messages. In *Proceedings of IAAI*, pages 1058–1065.
- Ryoko Tokuhsa, Kentaro Inui, and Yuji Matsumoto. 2008. Emotion classification using massive examples extracted from the Web. In *Proceedings of COLING*, pages 881–888.
- Joseph Weizenbaum. 1966. ELIZA — a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

# Utterance-Level Multimodal Sentiment Analysis

Verónica Pérez-Rosas and Rada Mihalcea

Computer Science and Engineering  
University of North Texas

veronicaperezrosas@my.unt.edu, rada@cs.unt.edu

Louis-Philippe Morency

Institute for Creative Technologies  
University of Southern California

morency@ict.usc.edu

## Abstract

During real-life interactions, people are naturally gesturing and modulating their voice to emphasize specific points or to express their emotions. With the recent growth of social websites such as YouTube, Facebook, and Amazon, video reviews are emerging as a new source of multimodal and natural opinions that has been left almost untapped by automatic opinion analysis techniques. This paper presents a method for multimodal sentiment classification, which can identify the sentiment expressed in utterance-level visual datastreams. Using a new multimodal dataset consisting of sentiment annotated utterances extracted from video reviews, we show that multimodal sentiment analysis can be effectively performed, and that the joint use of visual, acoustic, and linguistic modalities can lead to error rate reductions of up to 10.5% as compared to the best performing individual modality.

## 1 Introduction

Video reviews represent a growing source of consumer information that gained increasing interest from companies, researchers, and consumers. Popular web platforms such as YouTube, Amazon, Facebook, and ExpoTV have reported a significant increase in the number of consumer reviews in video format over the past five years. Compared to traditional text reviews, video reviews provide a more natural experience as they allow the viewer to better sense the reviewer's emotions, beliefs, and intentions through richer channels such as intonations, facial expressions, and body language.

Much of the work to date on opinion analysis has focused on textual data, and a number of resources have been created including lexicons (Wiebe and

Riloff, 2005; Esuli and Sebastiani, 2006) or large annotated datasets (Maas et al., 2011). Given the accelerated growth of other media on the Web and elsewhere, which includes massive collections of videos (e.g., YouTube, Vimeo, VideoLectures), images (e.g., Flickr, Picasa), audio clips (e.g., podcasts), the ability to address the identification of opinions in the presence of diverse modalities is becoming increasingly important. This has motivated researchers to start exploring multimodal clues for the detection of sentiment and emotions in video content (Morency et al., 2011; Wagner et al., 2011).

In this paper, we explore the addition of speech and visual modalities to text analysis in order to identify the sentiment expressed in video reviews. Given the non homogeneous nature of full-video reviews, which typically include a mixture of positive, negative, and neutral statements, we decided to perform our experiments and analyses at the utterance level. This is in line with earlier work on text-based sentiment analysis, where it has been observed that full-document reviews often contain both positive and negative comments, which led to a number of methods addressing opinion analysis at sentence level. Our results show that relying on the joint use of linguistic, acoustic, and visual modalities allows us to better sense the sentiment being expressed as compared to the use of only one modality at a time.

Another important aspect of this paper is the introduction of a new multimodal opinion database annotated at the utterance level which is, to our knowledge, the first of its kind. In our work, this dataset enabled a wide range of multimodal sentiment analysis experiments, addressing the relative importance of modalities and individual features.

The following section presents related work in text-based sentiment analysis and audio-visual emotion recognition. Section 3 describes our new multimodal datasets with utterance-level sentiment annotations. Section 4 presents our multimodal sen-

timent analysis approach, including details about our linguistic, acoustic, and visual features. Our experiments and results on multimodal sentiment classification are presented in Section 5, with a detailed discussion and analysis in Section 6.

## 2 Related Work

In this section we provide a brief overview of related work in text-based sentiment analysis, as well as audio-visual emotion analysis.

### 2.1 Text-based Subjectivity and Sentiment Analysis

The techniques developed so far for subjectivity and sentiment analysis have focused primarily on the processing of text, and consist of either rule-based classifiers that make use of opinion lexicons, or data-driven methods that assume the availability of a large dataset annotated for polarity. These tools and resources have been already used in a large number of applications, including expressive text-to-speech synthesis (Alm et al., 2005), tracking sentiment timelines in on-line forums and news (Balog et al., 2006), analysis of political debates (Carvalho et al., 2011), question answering (Oh et al., 2012), conversation summarization (Carenini et al., 2008), and citation sentiment detection (Athar and Teufel, 2012).

One of the first lexicons used in sentiment analysis is the General Inquirer (Stone, 1968). Since then, many methods have been developed to automatically identify opinion words and their polarity (Hatzivassiloglou and McKeown, 1997; Turney, 2002; Hu and Liu, 2004; Taboada et al., 2011), as well as n-gram and more linguistically complex phrases (Yang and Cardie, 2012).

For data-driven methods, one of the most widely used datasets is the MPQA corpus (Wiebe et al., 2005), which is a collection of news articles manually annotated for opinions. Other datasets are also available, including two polarity datasets consisting of movie reviews (Pang and Lee, 2004; Maas et al., 2011), and a collection of newspaper headlines annotated for polarity (Strapparava and Mihalcea, 2007).

While difficult problems such as cross-domain (Blitzer et al., 2007; Li et al., 2012) or cross-language (Mihalcea et al., 2007; Wan, 2009; Meng et al., 2012) portability have been addressed, not much has been done in terms of extending the applicability of sentiment analysis to other modalities,

such as speech or facial expressions.

The only exceptions that we are aware of are the findings reported in (Somasundaran et al., 2006; Raaijmakers et al., 2008; Mairesse et al., 2012; Metze et al., 2009), where speech and text have been analyzed jointly for the purpose of subjectivity or sentiment identification, without, however, addressing other modalities such as visual cues; and the work reported in (Morency et al., 2011; Perez-Rosas et al., 2013), where multimodal cues have been used for the analysis of sentiment in product reviews, but where the analysis was done at the much coarser level of full videos rather than individual utterances as we do in our work.

### 2.2 Audio-Visual Emotion Analysis.

Also related to our work is the research done on emotion analysis. Emotion analysis of speech signals aims to identify the emotional or physical states of a person by analyzing his or her voice (Ververidis and Kotropoulos, 2006). Proposed methods for emotion recognition from speech focus both on what is being said and how is being said, and rely mainly on the analysis of the speech signal by sampling the content at utterance or frame level (Bitouk et al., 2010). Several researchers used prosody (e.g., pitch, speaking rate, Mel frequency coefficients) for speech-based emotion recognition (Polzin and Waibel, 1996; Tato et al., 2002; Ayadi et al., 2011).

There are also studies that analyzed the visual cues, such as facial expressions and body movements (Calder et al., 2001; Rosenblum et al., 1996; Essa and Pentland, 1997). Facial expressions are among the most powerful and natural means for human beings to communicate their emotions and intentions (Tian et al., 2001). Emotions can be also expressed unconsciously, through subtle movements of facial muscles such as smiling or eyebrow raising, often measured and described using the Facial Action Coding System (FACS) (Ekman et al., 2002).

De Silva et. al. (De Silva et al., 1997) and Chen et. al. (Chen et al., 1998) presented one of the early works that integrate both acoustic and visual information for emotion recognition. In addition to work that considered individual modalities, there is also a growing body of work concerned with multimodal emotion analysis (Silva et al., 1997; Sebe et al., 2006; Zhihong et al., 2009; Wollmer et al., 2010).

Utterance transcription	Label
En este color, creo que era el color frambuesa. In this color, I think it was raspberry	neu
Pinta hermosísimo. It looks beautiful.	pos
Sinceramente, con respecto a lo que pinta y a que son hidratante, si son muy hidratantes. Honestly, talking about how they looks and hydrates, yes they are very hydrant.	pos
Pero el problema de estos labiales es que cuando uno se los aplica, te dejan un gusto asqueroso en la boca. But the problem with those lipsticks is that when you apply them, they leave a very nasty taste	neg
Sinceramente, es no es que sea el olor sino que es mas bien el gusto. Honestly, is not the smell, it is the taste.	neg

Table 1: Sample utterance-level annotations. The labels used are: pos(itive), neg(ative), neu(tral).

More recently, two challenges have been organized focusing on the recognition of emotions using audio and visual cues (Schuller et al., 2011a; Schuller et al., 2011b), which included sub-challenges on audio-only, video-only, and audio-video, and drew the participation of many teams from around the world. Note however that most of the previous work on audio-visual emotion analysis has focused exclusively on the audio and video modalities, and did not consider textual features, as we do in our work.

### 3 MOUD: Multimodal Opinion Utterances Dataset

For our experiments, we created a dataset of utterances (named MOUD) containing product opinions expressed in Spanish.<sup>1</sup> We chose to work with Spanish because it is a widely used language, and it is the native language of the main author of this paper.

We started by collecting a set of videos from the social media web site YouTube, using several keywords likely to lead to a product review or recommendation. Starting with the YouTube search page, videos were found using the following keywords: *mis products favoritos* (my favorite products), *products que no recomiendo* (non recommended products), *mis perfumes favoritos* (my favorite perfumes), *películas recomendadas* (recommended movies), *películas que no recomiendo* (non recommended movies) and *libros recomendados* (recommended books), *libros que no recomiendo* (non recommended books). Notice that the keywords are not targeted at a specific product type; rather, we used a variety of product names, so that the dataset has some degree of generality within the broad domain of product reviews.

<sup>1</sup>Publicly available from the authors webpage.

Among all the videos returned by the YouTube search, we selected only videos that respected the following guidelines: the speaker should be in front of the camera; her face should be clearly visible, with a minimum amount of face occlusion during the recording; there should not be any background music or animation. The final video set includes 80 videos randomly selected from the videos retrieved from YouTube that also met the guidelines above. The dataset includes 15 male and 65 female speakers, with their age approximately ranging from 20 to 60 years.

All the videos were first pre-processed to eliminate introductory titles and advertisements. Since the reviewers often switched topics when expressing their opinions, we manually selected a 30 seconds opinion segment from each video to avoid having multiple topics in a single review.

#### 3.1 Segmentation and Transcription

All the video clips were manually processed to transcribe the verbal statements and also to extract the start and end time of each utterance. Since the reviewers utter expressive sentences that are naturally segmented by speech pauses, we decided to use these pauses ( $>0.5$ seconds) to identify the beginning and the end of each utterance. The transcription and segmentation were performed using the Transcriber software.

Each video was segmented into an average of six utterances, resulting in a final dataset of 498 utterances. Each utterance is linked to the corresponding audio and video stream, as well as its manual transcription. The utterances have an average duration of 5 seconds, with a standard deviation of 1.2 seconds.

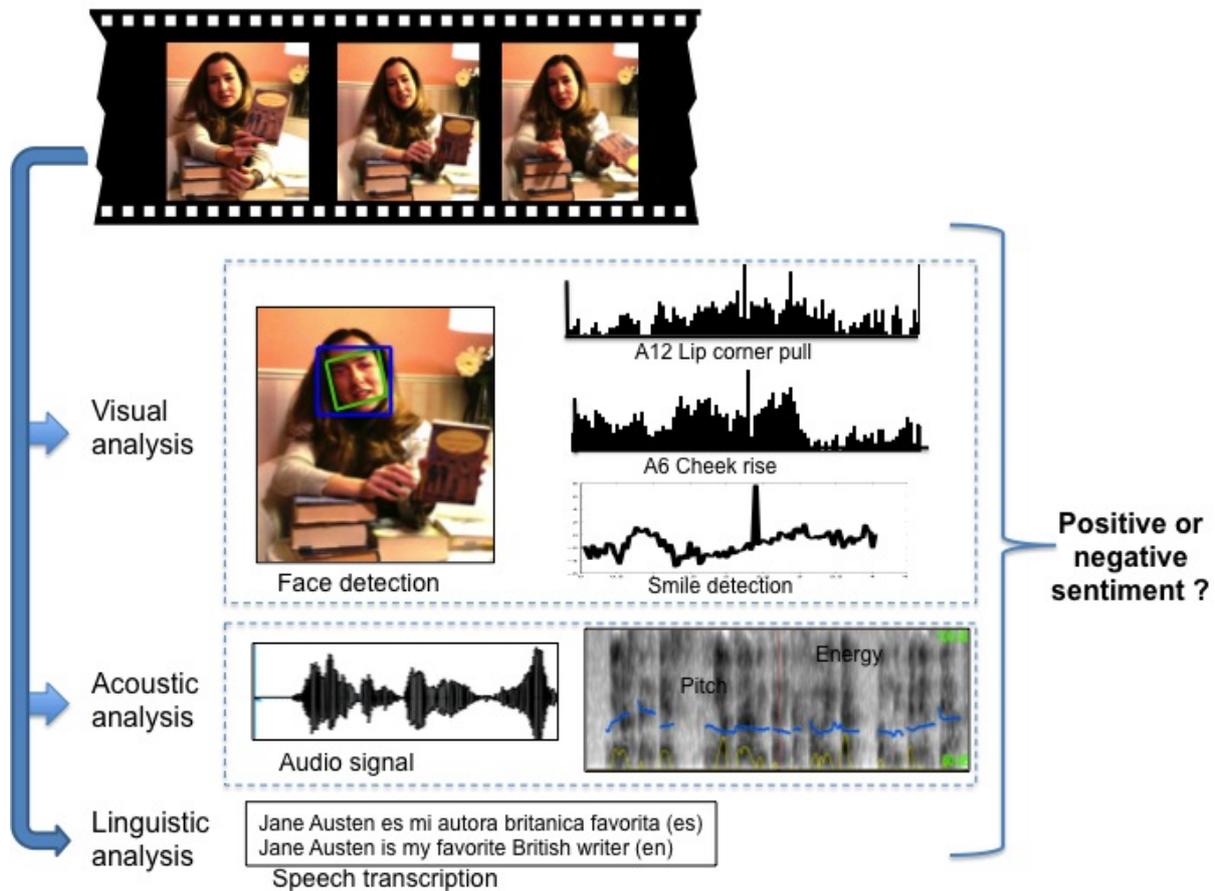


Figure 1: Multimodal feature extraction

### 3.2 Sentiment Annotation

To enable the use of this dataset for sentiment detection, we performed sentiment annotations at utterance level. Annotations were done using Elan,<sup>2</sup> which is a widely used tool for the annotation of video and audio resources. Two annotators independently labeled each utterance as positive, negative, or neutral. The annotation was done after seeing the video corresponding to an utterance (along with the corresponding audio source). The transcription of the utterance was also made available. Thus, the annotation process included all three modalities: visual, acoustic, and linguistic. The annotators were allowed to watch the video segment and their corresponding transcription as many times as needed.

The inter-annotator agreement was measured at 88%, with a Kappa of 0.81, which represents good agreement. All the disagreements were reconciled through discussions.

Table 1 shows the five utterances obtained from a video in our dataset, along with their corresponding

<sup>2</sup><http://tla.mpi.nl/tools/tla-tools/elan/>

sentiment annotations. As this example illustrates, a video can contain a mix of positive, negative, and neutral utterances. Note also that sentiment is not always explicit in the text: for example, the last utterance “Honestly, it is not the smell, it is the taste” has an implicit reference to the “nasty taste” expressed in the previous utterance, and thus it was also labeled as negative by both annotators.

## 4 Multimodal Sentiment Analysis

The main advantage that comes with the analysis of video opinions, as compared to their textual counterparts, is the availability of visual and speech cues. In textual opinions, the only source of information consists of words and their dependencies, which may sometime prove insufficient to convey the exact sentiment of the user. Instead, video opinions naturally contain multiple modalities, consisting of visual, acoustic, and linguistic datastreams. We hypothesize that the simultaneous use of these three modalities will help create a better opinion analysis model.

## 4.1 Feature Extraction

This section describes the process of automatically extracting linguistic, acoustic and visual features from the video reviews. First, we obtain the stream corresponding to each modality, followed by the extraction of a representative set of features for each modality, as described in the following subsections. These features are then used as cues to build a classifier of positive or negative sentiment. Figure 1 illustrates this process.

### 4.1.1 Linguistic Features

We use a bag-of-words representation of the video transcriptions of each utterance to derive unigram counts, which are then used as linguistic features. First, we build a vocabulary consisting of all the words, including stopwords, occurring in the transcriptions of the training set. We then remove those words that have a frequency below 10 (value determined empirically on a small development set). The remaining words represent the unigram features, which are then associated with a value corresponding to the frequency of the unigram inside each utterance transcription. These simple weighted unigram features have been successfully used in the past to build sentiment classifiers on text, and in conjunction with Support Vector Machines (SVM) have been shown to lead to state-of-the-art performance (Maas et al., 2011).

### 4.1.2 Acoustic Features

Acoustic features are automatically extracted from the speech signal of each utterance. We used the open source software OpenEAR (Schuller, 2009) to automatically compute a set of acoustic features. We include prosody, energy, voicing probabilities, spectrum, and cepstral features.

- Prosody features. These include intensity, loudness, and pitch that describe the speech signal in terms of amplitude and frequency.
- Energy features. These features describe the human loudness perception.
- Voice probabilities. These are probabilities that represent an estimate of the percentage of voiced and unvoiced energy in the speech.
- Spectral features. The spectral features are based on the characteristics of the human ear, which uses a nonlinear frequency unit to simulate the human auditory system. These features describe the speech formants, which

model spoken content and represent speaker characteristics.

- Cepstral features. These features emphasize changes or periodicity in the spectrum features measured by frequencies; we model them using 12 Mel-frequency cepstral coefficients that are calculated based on the Fourier transform of a speech frame.

Overall, we have a set of 28 acoustic features. During the feature extraction, we use a frame sampling of 25ms. Speaker normalization is performed using z-standardization. The voice intensity is thresholded to identify samples with and without speech, with the same threshold being used for all the experiments and all the speakers. The features are averaged over all the frames in an utterance, to obtain one feature vector for each utterance.

### 4.1.3 Facial Features

Facial expressions can provide important clues for affect recognition, which we use to complement the linguistic and acoustic features extracted from the speech stream.

The most widely used system for measuring and describing facial behaviors is the Facial Action Coding System (FACS), which allows for the description of face muscle activities through the use of a set of Action Units (AUs). According with (Ekman, 1993), there are 64 AUs that involve the upper and lower face, including several face positions and movements.<sup>3</sup> AUs can occur either by themselves or in combination, and can be used to identify a variety of emotions. While AUs are frequently annotated by certified human annotators, automatic tools are also available. In our work, we use the Computer Expression Recognition Toolbox (CERT) (Littlewort et al., 2011), which allows us to automatically extract the following visual features:

- Smile and head pose estimates. The smile feature is an estimate for smiles. Head pose detection consists of three-dimensional estimates of the head orientation, i.e., yaw, pitch, and roll. These features provide information about changes in smiles and face positions while uttering positive and negative opinions.
- Facial AUs. These features are the raw estimates for 30 facial AUs related to muscle movements for the eyes, eyebrows, nose, lips,

<sup>3</sup><http://www.cs.cmu.edu/afs/cs/project/face/www/facs.htm>

and chin. They provide detailed information about facial behaviors from which we expect to find differences between positive and negative states.

- Eight basic emotions. These are estimates for the following emotions: anger, contempt, disgust, fear, joy, sad, surprise, and neutral. These features describe the presence of two or more AUs that define a specific emotion. For example, the unit A12 describes the pulling of lip corners movement, which usually suggests a smile but when associated with a check raiser movement (unit A6), represents a marker for the emotion of happiness.

We extract a total of 40 visual features, each of them obtained at frame level. Since only one person is present in each video clip, most of the time facing the camera, the facial tracking was successfully applied for most of our data. For the analysis, we use a sampling rate of 30 frames per second. The features extracted for each utterance are averaged over all the valid frames, which are automatically identified using the output of CERT.<sup>4</sup> Segments with more than 60% of invalid frames are simply discarded.

## 5 Experiments and Results

We run our sentiment classification experiments on the MOUD dataset introduced earlier. From the dataset, we remove utterances labeled as neutral, thus keeping only the positive and negative utterances with valid visual features. The removal of neutral utterances is done for two main reasons. First, the number of neutral utterances in the dataset is rather small. Second, previous work in subjectivity and sentiment analysis has demonstrated that a layered approach (where neutral statements are first separated from opinion statements followed by a separation between positive and negative statements) works better than a single three-way classification. After this process, we are left with an experimental dataset of 412 utterances, 182 of which are labeled as positive, and 231 are labeled as negative.

From each utterance, we extract the linguistic, acoustic, and visual features described above, which are then combined using the early fusion (or feature-level fusion) approach (Hall and Llinas,

<sup>4</sup>There is a small number of frames that CERT could not process, mostly due to the brief occlusions that occur when the speaker is showing the product she is reviewing.

Modality	Accuracy
Baseline	55.93%
One modality at a time	
Linguistic	70.94%
Acoustic	64.85%
Visual	67.31%
Two modalities at a time	
Linguistic + Acoustic	72.88%
Linguistic + Visual	72.39%
Acoustic + Visual	68.86%
Three modalities at a time	
Linguistic+Acoustic+Visual	74.09%

Table 2: Utterance-level sentiment classification with linguistic, acoustic, and visual features.

1997; Atrey et al., 2010). In this approach, the features collected from all the multimodal streams are combined into a single feature vector, thus resulting in one vector for each utterance in the dataset which is used to make a decision about the sentiment orientation of the utterance.

We run several comparative experiments, using one, two, and three modalities at a time. We use the entire set of 412 utterances and run ten fold cross validations using an SVM classifier, as implemented in the Weka toolkit.<sup>5</sup> In line with previous work on emotion recognition in speech (Haq and Jackson, 2009; Anagnostopoulos and Vovoli, 2010) where utterances are selected in a speaker dependent manner (i.e., utterances from the same speaker are included in both training and test), as well as work on sentence-level opinion classification where document boundaries are not considered in the split performed between the training and test sets (Wilson et al., 2004; Wiegand and Klakow, 2009), the training/test split for each fold is performed at utterance level regardless of the video they belong to.

Table 2 shows the results of the utterance-level sentiment classification experiments. The baseline is obtained using the ZeroR classifier, which assigns the most frequent label by default, averaged over the ten folds.

## 6 Discussion

The experimental results show that sentiment classification can be effectively performed on multimodal datastreams. Moreover, the integration of

<sup>5</sup><http://www.cs.waikato.ac.nz/ml/weka/>

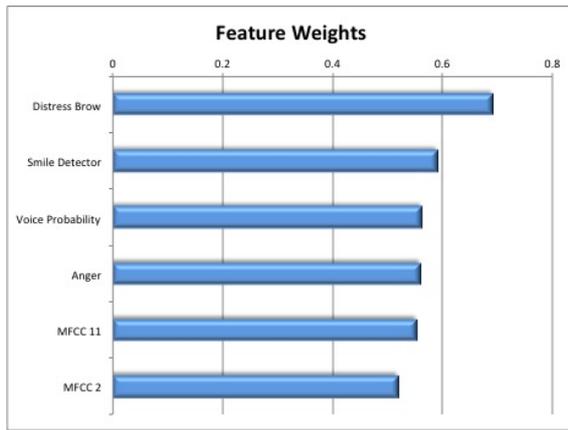


Figure 2: Visual and acoustic feature weights. This graph shows the relative importance of the information gain weights associated with the top most informative acoustic-visual features.

visual, acoustic, and linguistic features can improve significantly over the use of one modality at a time, with incremental improvements observed for each added modality.

Among the individual classifiers, the linguistic classifier appears to be the most accurate, followed by the classifier that relies on visual clues, and by the audio classifier. Compared to the best individual classifier, the relative error rate reduction obtained with the tri-modal classifier is 10.5%. The results obtained with this multimodal utterance classifier are found to be significantly better than the best individual results (obtained with the text modality), with significance being tested with a t-test ( $p=0.05$ ).

### Feature analysis.

To determine the role played by each of the visual and acoustic features, we compare the feature weights assigned by the learning algorithm, as shown in Figure 2. Interestingly, a distressed brow is the strongest indicator of sentiment, followed, this time not surprisingly, by the smile feature. Other informative features for sentiment classification are the voice probability, representing the energy in speech, the combined visual features that represent an angry face, and two of the cepstral coefficients.

To reach a better understanding of the relation between features, we also calculate the Pearson correlation between the visual and acoustic features. Table 3 shows a subset of these correlation figures. As we expected, correlations between features of the same type are higher. For example,

the correlation between features AU6 and AU12 or the correlation between intensity and loudness is higher than the correlation between AU6 and intensity. Nonetheless, we still find some significant correlations between features of different types, for instance AU12 and AU45 which are both significantly correlated with the intensity and loudness features. This give us confidence about using them for further analysis.

### Video-level sentiment analysis.

To understand the role played by the size of the video-segments considered in the sentiment classification experiments, as well as the potential effect of a speaker-independence assumption, we also run a set of experiments where we use full videos for the classification.

In these experiments, once again the sentiment annotation is done by two independent annotators, using the same protocol as in the utterance-based annotations. Videos that were ambivalent about the general sentiment were either labeled as neutral (and thus removed from the experiments), or labeled with the dominant sentiment. The inter-annotator agreement for this annotation was measured at 96.1%. As before, the linguistic, acoustic, and visual features are averaged over the entire video, and we use an SVM classifier in ten-fold cross validation experiments.

Table 4 shows the results obtained in these video-level experiments. While the combination of modalities still helps, the improvement is smaller than the one obtained during the utterance-level classification. Specifically, the combined effect of acoustic and visual features improves significantly over the individual modalities. However, the combination of linguistic features with other modalities does not lead to clear improvements. This may be due to the smaller number of feature vectors used in the experiments (only 80, as compared to the 412 used in the previous setup). Another possible reason is the fact that the acoustic and visual modalities are significantly weaker than the linguistic modality, most likely due to the fact that the feature vectors are now speaker-independent, which makes it harder to improve over the linguistic modality alone.

## 7 Conclusions

In this paper, we presented a multimodal approach for utterance-level sentiment classification. We introduced a new multimodal dataset consisting

	AU6	AU12	AU45	AUs 1,1+4	Pitch	Voice probability	Intensity	Loudness
AU6	1.00	0.46*	-0.03	-0.05	0.06	-0.14*	-0.04	-0.02
AU12		1.00	-0.23*	-0.33*	0.04	0.05	0.15*	0.16*
AU45			1.00	0.05	-0.05	-0.11*	-.163*	0.16*
AUs 1,1+4				1.00	-0.11*	-0.16*	0.06	0.07
Pitch					1.00	-0.04	-0.01	-0.08
Voice probability						1.00	0.19*	0.38*
Intensity							1.00	0.85*
Loudness								1.00

Table 3: Correlations between several visual and acoustic features. Visual features: AU6 Cheek raise, AU12 Lip corner pull, AU45 Blink eye and closure, AU1,1+4 Distress brow. Acoustic features: Pitch, Voice probability, Intensity, Energy. \*Correlation is significant at the 0.05 level (1-tailed)

Modality	Accuracy
Baseline	55.93%
One modality at a time	
Linguistic	73.33%
Acoustic	53.33%
Visual	50.66%
Two modalities at a time	
Linguistic + Acoustic	72.00%
Linguistic + Visual	74.66%
Acoustic + Visual	61.33%
Three modalities at a time	
Linguistic+Acoustic+Visual	74.66%

Table 4: Video-level sentiment classification with linguistic, acoustic, and visual features.

of sentiment annotated utterances extracted from video reviews, where each utterance is associated with a video, acoustic, and linguistic datastream. Our experiments show that sentiment annotation of utterance-level visual datastreams can be effectively performed, and that the use of multiple modalities can lead to error rate reductions of up to 10.5% as compared to the use of one modality at a time. In future work, we plan to explore alternative multimodal fusion methods, such as decision-level and meta-level fusion, to improve the integration of the visual, acoustic, and linguistic modalities.

## Acknowledgments

We would like to thank Alberto Castro for his help with the sentiment annotations. This material is based in part upon work supported by National Science Foundation awards #0917170 and #1118018, by DARPA-BAA-12-47 DEFT grant #12475008, and by a grant from U.S. RDECOM. Any opinions,

findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Defense Advanced Research Projects Agency, or the U.S. Army Research, Development, and Engineering Command.

## References

- C. Alm, D. Roth, and R. Sproat. 2005. Emotions from text: Machine learning for text-based emotion prediction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 347–354, Vancouver, Canada.
- C. Anagnostopoulos and E. Vovoli. 2010. Sound processing features for speaker-dependent and phrase-independent emotion recognition in berlin database. In *Information Systems Development*, pages 413–421. Springer.
- A. Athar and S. Teufel. 2012. Context-enhanced citation sentiment detection. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada, June.
- P. K. Atrey, M. A. Hossain, A. El Saddik, and M. Kankanhalli. 2010. Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems*, 16.
- M. El Ayadi, M. Kamel, and F. Karray. 2011. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44(3):572 – 587.
- K. Balog, G. Mishne, and M. de Rijke. 2006. Why are they excited? identifying and explaining spikes in blog mood levels. In *Proceedings of the 11th Meeting of the European Chapter of the Association for Computational Linguistics (EACL-2006)*.
- Dmitri Bitouk, Ragini Verma, and Ani Nenkova. 2010. Class-level spectral features for emotion recognition. *Speech Commun.*, 52(7-8):613–625, July.

- J. Blitzer, M. Dredze, and F. Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics*.
- A. J. Calder, A. M. Burton, P. Miller, A. W. Young, and S. Akamatsu. 2001. A principal component analysis of facial expressions. *Vision research*, 41(9):1179–1208, April.
- G. Carenini, R. Ng, and X. Zhou. 2008. Summarizing emails with conversational cohesion and subjectivity. In *Proceedings of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2008)*, Columbus, Ohio.
- P. Carvalho, L. Sarmiento, J. Teixeira, and M. Silva. 2011. Liars and saviors in a sentiment annotated corpus of comments to political debates. In *Proceedings of the Association for Computational Linguistics (ACL 2011)*, Portland, OR.
- L. S. Chen, T. S. Huang, T. Miyasato, and R. Nakatsu. 1998. Multimodal human emotion/expression recognition. In *Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, pages 366–, Washington, DC, USA. IEEE Computer Society.
- L C De Silva, T Miyasato, and R Nakatsu, 1997. *Facial emotion recognition using multi-modal information*, volume 1, page 397401. IEEE Signal Processing Society.
- P. Ekman, W. Friesen, and J. Hager. 2002. Facial action coding system.
- P. Ekman. 1993. Facial expression of emotion. *American Psychologist*, 48:384–392.
- I.A. Essa and A.P. Pentland. 1997. Coding, analysis, interpretation, and recognition of facial expressions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):757–763, jul.
- A. Esuli and F. Sebastiani. 2006. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC 2006)*, Genova, IT.
- D.L. Hall and J. Llinas. 1997. An introduction to multisensor fusion. *IEEE Special Issue on Data Fusion*, 85(1).
- S. Haq and P. Jackson. 2009. Speaker-dependent audio-visual emotion recognition. In *International Conference on Audio-Visual Speech Processing*.
- V. Hatzivassiloglou and K. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, pages 174–181.
- M. Hu and B. Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, Seattle, Washington.
- F. Li, S. J. Pan, O. Jin, Q. Yang, and X. Zhu. 2012. Cross-domain co-extraction of sentiment and topic lexicons. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju Island, Korea.
- G. Littlewort, J. Whitehill, Tingfan Wu, I. Fasel, M. Frank, J. Movellan, and M. Bartlett. 2011. The computer expression recognition toolbox (cert). In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 298–305, march.
- A. Maas, R. Daly, P. Pham, D. Huang, A. Ng, and C. Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the Association for Computational Linguistics (ACL 2011)*, Portland, OR.
- F. Mairesse, J. Polifroni, and G. Di Fabbrizio. 2012. Can prosody inform sentiment analysis? experiments on short spoken reviews. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 5093–5096, march.
- X. Meng, F. Wei, X. Liu, M. Zhou, G. Xu, and H. Wang. 2012. Cross-lingual mixture model for sentiment classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju Island, Korea.
- F. Metze, T. Polzehl, and M. Wagner. 2009. Fusion of acoustic and linguistic features for emotion detection. In *Semantic Computing, 2009. ICSC '09. IEEE International Conference on*, pages 153–160, sept.
- R. Mihalcea, C. Banea, and J. Wiebe. 2007. Learning multilingual subjective language via cross-lingual projections. In *Proceedings of the Association for Computational Linguistics*, Prague, Czech Republic.
- L.P. Morency, R. Mihalcea, and P. Doshi. 2011. Towards multimodal sentiment analysis: Harvesting opinions from the web. In *Proceedings of the International Conference on Multimodal Computing*, Alicante, Spain.
- J. Oh, K. Torisawa, C. Hashimoto, T. Kawada, S. De Saeger, J. Kazama, and Y. Wang. 2012. Why question answering using sentiment analysis and word classes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea.
- B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July.

- V. Perez-Rosas, R. Mihalcea, and L.-P. Morency. 2013. Multimodal sentiment analysis of spanish online videos. *IEEE Intelligent Systems*.
- T. Polzin and A. Waibel. 1996. Recognizing emotions in speech. In *In ICSLP*.
- S. Raaijmakers, K. Truong, and T. Wilson. 2008. Multimodal subjectivity analysis of multiparty conversation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 466–474, Honolulu, Hawaii.
- M. Rosenblum, Y. Yacoob, and L.S. Davis. 1996. Human expression recognition from motion using a radial basis function network architecture. *Neural Networks, IEEE Transactions on*, 7(5):1121–1138, sep.
- B. Schuller, M. Valstar, R. Cowie, and M. Pantic, editors. 2011a. *Audio/Visual Emotion Challenge and Workshop (AVEC 2011)*.
- B. Schuller, M. Valstar, F. Eyben, R. Cowie, and M. Pantic, editors. 2011b. *Audio/Visual Emotion Challenge and Workshop (AVEC 2011)*.
- F. Eyben M. Wollmer B. Schuller. 2009. Openear introducing the munich open-source emotion and affect recognition toolkit. In *ACII*.
- N. Sebe, I. Cohen, T. Gevers, and T.S. Huang. 2006. Emotion recognition based on joint visual and audio cues. In *ICPR*.
- D. Silva, T. Miyasato, and R. Nakatsu. 1997. Facial emotion recognition using multi-modal information. In *Proceedings of the International Conference on Information and Communications Security*.
- S. Somasundaran, J. Wiebe, P. Hoffmann, and D. Litman. 2006. Manual annotation of opinion categories in meetings. In *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*.
- P. Stone. 1968. *General Inquirer: Computer Approach to Content Analysis*. MIT Press.
- C. Strapparava and R. Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the 4th International Workshop on the Semantic Evaluations (SemEval 2007)*, Prague, Czech Republic.
- M. Taboada, J. Brooke, M. Tofiloski, K. Voli, and M. Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(3).
- R. Tato, R. Santos, R. Kompe, and J. M. Pardo. 2002. Emotional space improves emotion recognition. In *In Proc. ICSLP 2002*, pages 2029–2032.
- Y.-I. Tian, T. Kanade, and J.F. Cohn. 2001. Recognizing action units for facial expression analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(2):97–115, feb.
- P. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 417–424, Philadelphia.
- D. Ververidis and C. Kotropoulos. 2006. Emotional speech recognition: Resources, features, and methods. *Speech Communication*, 48(9):1162–1181, September.
- J. Wagner, E. Andre, F. Lingenfeller, and Jonghwa Kim. 2011. Exploring fusion methods for multimodal emotion recognition with missing data. *Affective Computing, IEEE Transactions on*, 2(4):206–218, oct.-dec.
- X. Wan. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint Conference of the Association of Computational Linguistics and the International Joint Conference on Natural Language Processing*, Singapore, August.
- J. Wiebe and E. Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2005) (invited paper)*, Mexico City, Mexico.
- J. Wiebe, T. Wilson, and C. Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.
- M. Wiegand and D. Klakow. 2009. The role of knowledge-based features in polarity classification at sentence level. In *Proceedings of the International Conference of the Florida Artificial Intelligence Research Society*.
- T. Wilson, J. Wiebe, and R. Hwa. 2004. Just how mad are you? finding strong and weak opinion clauses. In *Proceedings of the American Association for Artificial Intelligence*.
- M. Wollmer, B. Schuller, F. Eyben, and G. Rigoll. 2010. Combining long short-term memory and dynamic bayesian networks for incremental emotion-sensitive artificial listening. *IEEE Journal of Selected Topics in Signal Processing*, 4(5), October.
- B. Yang and C. Cardie. 2012. Extracting opinion expressions with semi-markov conditional random fields. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea.
- Z. Zhihong, M. Pantic G.I. Roisman, and T.S. Huang. 2009. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *PAMI*, 31(1).

# Probabilistic Sense Sentiment Similarity through Hidden Emotions

Mitra Mohtarami<sup>1</sup>, Man Lan<sup>2</sup>, and Chew Lim Tan<sup>1</sup>

<sup>1</sup>Department of Computer Science, National University of Singapore;

<sup>2</sup>Department of Computer Science, East China Normal University

{mitra, tancl}@comp.nus.edu.sg;mlan@cs.ecnu.edu.cn

## Abstract

*Sentiment Similarity* of word pairs reflects the distance between the words regarding their underlying sentiments. This paper aims to infer the sentiment similarity between word pairs with respect to their senses. To achieve this aim, we propose a probabilistic emotion-based approach that is built on a hidden emotional model. The model aims to predict a vector of basic human emotions for each sense of the words. The resultant emotional vectors are then employed to infer the sentiment similarity of word pairs. We apply the proposed approach to address two main NLP tasks, namely, *Indirect yes/no Question Answer Pairs* inference and *Sentiment Orientation* prediction. Extensive experiments demonstrate the effectiveness of the proposed approach.

## 1 Introduction

Sentiment similarity reflects the distance between words based on their underlying sentiments. Semantic similarity measures such as Latent Semantic Analysis (LSA) (Landauer et al., 1998) can effectively capture the similarity between semantically related words like "car" and "automobile", but they are less effective in relating words with similar sentiment orientation like "excellent" and "superior". For example, the following relations show the semantic similarity between some sentiment words computed by LSA:

$$\begin{aligned} \mathbf{E1}: \text{LSA}(\text{excellent}, \text{superior}) &= 0.40 \\ &< \text{LSA}(\text{excellent}, \text{good}) = 0.46 \\ &< \text{LSA}(\text{good}, \text{bad}) = 0.65 \end{aligned}$$

Clearly, the sentiment similarity between the above words should be in the reversed order. In fact, the sentiment intensity in "excellent" is closer to "superior" than "good". Furthermore,

sentiment similarity between "good" and "bad" should be 0.

In this paper, we propose a probabilistic approach to detect the sentiment similarity of words regarding their senses and underlying sentiments. For this purpose, we propose to model the hidden emotions of word senses. We show that our approach effectively outperforms the semantic similarity measures in two NLP tasks: *Indirect yes/no Question Answer Pairs (IQAPs) Inference* and *Sentiment Orientation (SO) prediction* that are described as follows:

In IQAPs, answers do not explicitly contain the *yes* or *no* keywords, but rather provide context information to infer the *yes* or *no* answer (e.g. **Q:** *Was she the **best** one on that old show?* **A:** *She was simply **funny**.*). Clearly, the sentiment words in IQAPs are the pivots to infer the *yes* or *no* answers. We show that sentiment similarity between such words (e.g., here the adjectives *best* and *Funny*) can be used effectively to infer the answers.

The second application (SO prediction) aims to determine the sentiment orientation of individual words. Previous research utilized the semantic relations between words obtained from WordNet (Hassan and Radev, 2010) and semantic similarity measures (e.g. Turney and Littman, 2003) for this purpose. In this paper, we show that sentiment similarity between word pairs can be effectively utilized to compute SO of words.

The contributions of this paper are follows:

- We propose an effective approach to predict the sentiment similarity between word pairs through hidden emotions at the sense level,
- We show the utility of sentiment similarity prediction in IQAP inference and SO prediction tasks, and
- Our hidden emotional model can infer the type and number of hidden emotions in a corpus.

## 2 Sentiment Similarity through Hidden Emotions

As we discussed above, semantic similarity measures are less effective to infer sentiment similarity between word pairs. In addition, different senses of sentiment words carry different human emotions. In fact, a sentiment word can be represented as a vector of emotions with *intensity* values from "very weak" to "very strong". For example, Table 1 shows several sentiment words and their corresponding emotion vectors based the following set of emotions:  $e = [\textit{anger}, \textit{disgust}, \textit{sadness}, \textit{fear}, \textit{guilt}, \textit{interest}, \textit{joy}, \textit{shame}, \textit{surprise}]$ . For example, "deceive" has 0.4 and 0.5 intensity values with respect to the emotions "disgust" and "sadness" with an overall -0.9 (i.e.  $-0.4-0.5$ ) value for sentiment orientation (Neviarouskaya et al., 2007; Neviarouskaya et al., 2009).

Word	Emotional Vector	SO
$e = [\textit{anger}, \textit{disgust}, \textit{sadness}, \textit{fear}, \textit{guilt}, \textit{interest}, \textit{joy}, \textit{shame}, \textit{surprise}]$		
<i>Rude</i>	[0.2, 0.4, 0, 0, 0, 0, 0, 0]	-0.6
<i>doleful</i>	[0, 0, 0.4, 0, 0, 0, 0, 0]	-0.4
<i>smashed</i>	[0, 0, 0.8, 0.6, 0, 0, 0, 0]	-1.4
<i>shamefully</i>	[0, 0, 0, 0, 0, 0, 0.7, 0]	-0.7
<i>deceive</i>	[0, 0.4, 0.5, 0, 0, 0, 0, 0]	-0.9

Table 1. Sample of emotional vectors

The difficulty of the sentiment similarity prediction task is evident when terms carry different types of emotions. For instance, all the words in Table 1 have negative sentiment orientation, but, they carry different emotions with different emotion vectors. For example, "rude" reflects the emotions "anger" and "disgust", while the word "doleful" only reflects the emotion "sadness". As such, the word "doleful" is closer to the words "smashed" and "deceive" involving the emotion "sadness" than others. We show that emotion vectors of the words can be effectively utilized to predict the sentiment similarity between them.

Previous research shows little agreement about the number and types of the basic emotions (Ortony and Turner 1990; Izard 1971). Thus, we assume that the number and types of basic emotions are hidden and not pre-defined and propose a Probabilistic Sense Sentiment Similarity (PSSS) approach to extract the hidden emotions of word senses to infer their sentiment similarity.

## 3 Hidden Emotional Model

Online review portals provide rating mechanisms (in terms of stars, e.g. 5- or 10-star rating) to al-

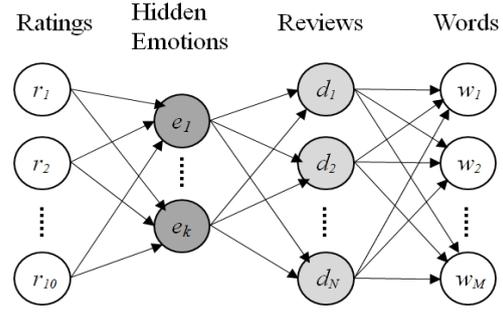


Figure 1. The structure of PSSS model

low users to attach ratings to their reviews. A rating indicates the summarized opinion of a user who ranks a product or service based on his feelings. There are various feelings and emotions behind such ratings with respect to the content of the reviews.

Figure 1 shows the intermediate layer of hidden emotions behind the ratings (sentiments) assigned to the documents (reviews) containing the words. This Figure indicates the general structure of our PSSS model. It shows that hidden emotions ( $e_i$ ) link the rating ( $r_j$ ) and the documents ( $d_k$ ). In this Section, we aim to employ ratings and the relations among ratings, documents, and words to extract the hidden emotions.

Figure 2 illustrates a simple graphical model using plate representation of Figure 1. As Figure 2 shows, the rating  $r$  from a set of ratings  $R = \{r_1, \dots, r_p\}$  is assigned to a hidden emotion set  $E = \{e_1, \dots, e_k\}$ . A document  $d$  from a set of documents  $D = \{d_1, \dots, d_N\}$  with vocabulary set  $W = \{w_1, \dots, w_M\}$  is associated with the hidden emotion set.

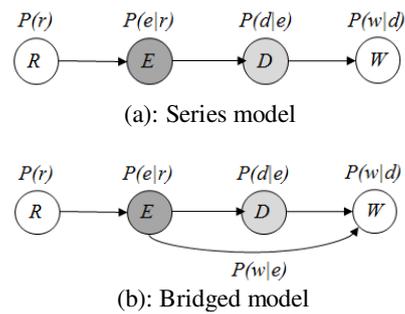


Figure 2. Hidden emotional model

The model presented in Figure 2(a) has been explored in (Mohtarami et al., 2013) and is called *Series Hidden Emotional Model* (SHEM). This representation assumes that *the word  $w$  is dependent to  $d$  and independent to  $e$*  (we refer to this Assumption as *A1*). However, in reality, a word  $w$  can inherit properties (e.g., emotions)

from the document  $d$  that contains  $w$ . Thus, we can assume that  $w$  is implicitly dependant on  $e$ . To account for this, we present *Bridged Hidden Emotional Model* (BHEM) shown in Figure 2(b). Our assumption, A2, in the BHEM model is as follows:  $w$  is dependent to both  $d$  and  $e$ .

Considering Figure 1, we represent the entire text collection as a set of  $(w,d,r)$  in which each observation  $(w,d,r)$  is associated with a set of unobserved emotions. If we assume that the observed tuples are independently generated, the whole data set is generated based on the joint probability of the observation tuples  $(w,d,r)$  as the follows (Mohtarami et al., 2013):

$$D = \prod_r \prod_d \prod_w P(w,d,r)^{n(w,d,r)} = \prod_r \prod_d \prod_w P(w,d,r)^{n(w,d)n(d,r)} \quad (1)$$

where,  $P(w,d,r)$  is the joint probability of the tuple  $(w,d,r)$ , and  $n(w,d,r)$  is the frequency of  $w$  in document  $d$  of rating  $r$  (note that  $n(w,d)$  is the term frequency of  $w$  in  $d$  and  $n(d,r)$  is one if  $r$  is assigned to  $d$ , and 0 otherwise). The joint probability for the BHEM is defined as follows considering hidden emotion  $e$ :

- regarding class probability of the hidden emotion  $e$  to be assigned to the observation  $(w,d,r)$ :

$$\begin{aligned} P(w,d,r) &= \sum_e P(w,d,r|e)P(e) = \\ &= \sum_e P(w,d|e)P(r|e)P(e) \\ - \text{regarding assumption A2 and Bayes' Rule:} \\ &= \sum_e P(w|d,e)P(d,e)P(r|e) \\ - \text{using Bayes' Rule:} \\ &= \sum_e P(d,e|w)P(w)P(r|e) \\ - \text{regarding A2 and conditional independency:} \\ &= \sum_e P(d|w)P(e|w)P(w)P(r|e) \\ &= P(d|w) \sum_e P(w|e)P(e)P(r|e) \end{aligned} \quad (2)$$

In the bridged model, the joint probability does not depend on the probability  $P(d|e)$  and the probabilities  $P(w|e)$ ,  $P(e)$  and  $P(r|e)$  are unknown, while in the SHEM model explained in (Mohtarami et al., 2013), the joint probability does not depend on  $P(w|e)$ , and probabilities  $P(d|e)$ ,  $P(e)$ , and  $P(r|e)$  are unknown.

We employ Maximum Likelihood approach to learn the probabilities and infer the possible hidden emotions. The log-likelihood of the whole

data set  $D$  in Equation (1) can be defined as follows:

$$L = \sum_r \sum_d \sum_w n(w,d)n(d,r) \log P(w,d,r) \quad (3)$$

Replacing  $P(w,d,r)$  by the values computed using the bridged model in Equation (2) results in:

$$L = \sum_r \sum_d \sum_w n(w,d)n(d,r) \log [P(d|w) \sum_e P(w|e)P(e)P(r|e)] \quad (4)$$

The above optimization problems are hard to compute due to the log of sum. Thus, *Expectation-maximization* (EM) is usually employed. EM consists of two following steps:

1. **E-step**: Calculates posterior probabilities for hidden emotions given the words, documents and ratings, and
2. **M-step**: Updates unknown probabilities (such as  $P(w|e)$  etc) using the posterior probabilities in the E-step.

The steps of EM can be computed for BHEM model. EM of the model employs assumptions A2 and Bayes Rule and is defined as follows:

**E-step**:

$$P(e|w,d,r) = \frac{P(r|e)P(e)P(w|e)}{\sum_e P(r|e)P(e)P(w|e)} \quad (5)$$

**M-step**:

$$\begin{aligned} P(r|e) &= \frac{\sum_d \sum_w n(w,d)n(d,r) P(e|w,d,r)}{\sum_r \sum_d \sum_w n(w,d)n(d,r) P(e|w,d,r)} \\ &= \frac{\sum_w n(w,r) P(e|w,d,r)}{\sum_r \sum_w n(w,r) P(e|w,d,r)} \end{aligned} \quad (6)$$

$$\begin{aligned} P(w|e) &= \frac{\sum_r \sum_d n(w,d)n(d,r) P(e|w,d,r)}{\sum_w \sum_r \sum_d n(w,d)n(d,r) P(e|w,d,r)} \\ &= \frac{\sum_r n(w,r) P(e|w,d,r)}{\sum_w \sum_r n(w,r) P(e|w,d,r)} \end{aligned} \quad (7)$$

$$\begin{aligned} P(e) &= \frac{\sum_r \sum_d \sum_w n(w,d)n(d,r) P(e|w,d,r)}{\sum_e \sum_d \sum_r \sum_w n(w,d)n(d,r) P(e|w,d,r)} \\ &= \frac{\sum_r \sum_w n(w,r) P(e|w,d,r)}{\sum_e \sum_r \sum_w n(w,r) P(e|w,d,r)} \end{aligned} \quad (8)$$

Note that in Equation (5), the probability  $P(e|w,d,r)$  does not depend on the document  $d$ . Also, in Equations (6)-(8) we remove the dependency on document  $d$  using the following Equation:

$$\sum_d n(w,d)n(d,r) = n(w,r) \quad (9)$$

where  $n(w,r)$  is the occurrence of  $w$  in all the documents in the rating  $r$ .

The EM steps computed by the bridged model do not depend on the variable document  $d$ , and discard  $d$  from the model. The reason is that  $w$  bypasses  $d$  to directly associate with the hidden emotion  $e$  in Figure 2(b).

Similar to BHEM, the EM steps for SHEM can be computed by considering assumptions *AI* and Bayes Rule as follows (Mohtarami et al., 2013):

**E-step:**

$$P(e|w, d, r) = \frac{P(r|e)P(e)P(d|e)}{\sum_e P(r|e)P(e)P(d|e)} \quad (10)$$

**M-step:**

$$P(r|e) = \frac{\sum_d \sum_w n(w, d)n(d, r)P(e|w, d, r)}{\sum_r \sum_d \sum_w n(w, d)n(d, r)P(e|w, d, r)} \quad (11)$$

$$P(d|e) = \frac{\sum_r \sum_w n(w, d)n(d, r)P(e|w, d, r)}{\sum_d \sum_r \sum_w n(w, d)n(d, r)P(e|w, d, r)} \quad (12)$$

$$P(e) = \frac{\sum_r \sum_d \sum_w n(w, d)n(d, r)P(e|w, d, r)}{\sum_e \sum_d \sum_r \sum_w n(w, d)n(d, r)P(e|w, d, r)} \quad (13)$$

Finally, we construct the emotional vectors using the algorithm presented in Table 2. The algorithm employs document-rating, term-document and term-rating matrices to infer the unknown probabilities. This algorithm can be used with both bridged or series models. Our goal is to infer the emotional vector for each word  $w$  that can be obtained by the probability  $P(w|e)$ . Note that, this probability can be simply computed for the SHEM model using  $P(d|e)$  as follows:

$$P(w|e) = \sum_d P(w|d)P(d|e) \quad (14)$$

### 3.1 Enriching Hidden Emotional Models

We enrich our emotional model by employing the requirement that the emotional vectors of two synonym words  $w_1$  and  $w_2$  should be similar. For this purpose, we utilize the semantic similarity between each two words and create an enriched matrix. Equation (15) shows how we compute this matrix. To compute the semantic similarity between word senses, we utilize their synsets as follows:

$$\begin{aligned} w_i w_j &= P(\text{syn}(w_i) | \text{syn}(w_j)) \\ &= \frac{1}{|\text{syn}(w_i)|} \sum_{\text{syn}(w_i)} \frac{1}{|\text{syn}(w_j)|} \sum_{\text{syn}(w_j)} P(w_i | w_j) \end{aligned} \quad (15)$$

where,  $\text{syn}(w)$  is the synset of  $w$ . Let  $\text{count}(w_i, w_j)$  be the co-occurrence of the  $w_i$  and  $w_j$ , and let  $\text{count}(w_j)$  be the total word count. The probability of  $w_i$  given  $w_j$  will then be  $P(w_i | w_j) = \text{count}(w_i, w_j) / \text{count}(w_j)$ . In addition, note that employing the synset of the words help to obtain different emotional vectors for each sense of a word.

The resultant enriched matrix  $W \times W$  is multiplied to the inputs of our hidden model (matrices  $W \times D$  or  $W \times R$ ). Note that this takes into account

**Input:**

*Series Model:* Document-Rate  $D \times R$ , Term-Document  $W \times D$   
*Bridged Model:* Term-Rate  $W \times R$

**Output:** Emotional vectors  $\{e_1, e_2, \dots, e_k\}$  for  $w$

**Algorithm:**

1. Enriching hidden emotional model:

*Series Model:* Update Term-Document  $W \times D$   
*Bridged Model:* Update Term-Rate  $W \times R$

2. Initialize unknown probabilities:

*Series Model:* Initialize  $P(d|e)$ ,  $P(r|e)$ , and  $P(e)$ , randomly  
*Bridged Model:* Initialize  $P(w|e)$ ,  $P(r|e)$ , and  $P(e)$

3. **while**  $L$  has not converged to a pre-specified value **do**

4. E-step;

*Series Model:* estimate the value of  $P(e|w, d, r)$  in Equation 10  
*Bridged Model:* estimate the value of  $P(e|w, d, r)$  in Equation 5

5. M-step;

*Series Model:* estimate the values of  $P(r|e)$ ,  $P(d|e)$ , and  $P(e)$  in Equations 11-13, respectively  
*Bridged Model:* estimate the values of  $P(r|e)$ ,  $P(w|e)$ , and  $P(e)$  in Equations 6-8, respectively

6. **end while**

7. **If** series hidden emotional model is used then

8. Infer word emotional vector: estimate  $P(w|e)$  in Equation 14.

9. **End if**

Table 2. Constructing emotional vectors via  $P(w|e)$

the senses of the words as well. The learning step of EM is done using the updated inputs. In this case, the correlated words can inherit the properties of each other. For example, if  $w_i$  does not occur in a document or rating involving another word (i.e.,  $w_j$ ), the word  $w_i$  can still be indirectly associated with the document or rating through the word  $w_j$ . However, the distribution of the opinion words in documents and ratings is not uniform. This may decrease the effectiveness of the enriched matrix.

The nonuniform distribution of opinion words has been also reported by Amiri et al. (2012) who showed that the positive words are frequently used in negative reviews. We also observed the same pattern in the development dataset. Figure 3 shows the overall occurrence of some positive and negative seeds in various ratings. As shown, in spite of the negative words, the positive words may frequently occur in both positive and negative documents. Such distribution of

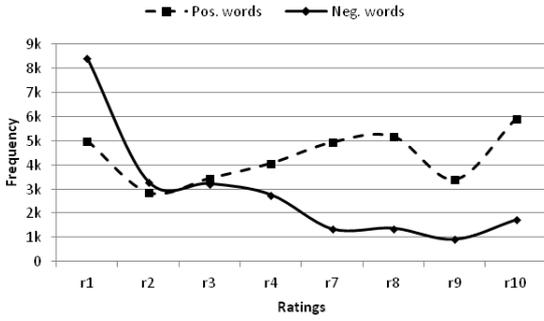


Figure 3. Nonuniform distribution of opinion words

positive words can mislead the enriched model.

To address this issue, we measure the confidence of an opinion word in the enriched matrix as follows.

$$Confidence_w = \frac{ABS[(TF_w^- \times DF_w^-) - (TF_w^+ \times DF_w^+)]}{(TF_w^- \times DF_w^-) + (TF_w^+ \times DF_w^+)} \quad (16)$$

where,  $TF_w^-$  ( $TF_w^+$ ) is the frequency of  $w$  in the ratings 1 to 4 (7 to 10), and  $DF_w^-$  ( $DF_w^+$ ) is the total number of documents with rating 1 to 4 (7 to 10) that contain  $w$ . The confidence value of  $w$  varies from 0 to 1, and it increases if:

- There is a large difference between the occurrences of  $w$  in positive and negative ratings.
- There is a large number of reviews involving  $w$  in the relative ratings.

To improve the efficiency of enriched matrix, the columns corresponding to each word in the matrix are multiplied by its confidence value.

#### 4 Predicting Sentiment Similarity

We utilize the approach proposed in (Mohtarami et al., 2013) to compute the sentiment similarity between two words. This approach compares the emotional vector of the given words. Let  $X$  and  $Y$  be the emotional vectors of two words. Equation (17) computes their correlation:

$$corr(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)S_X S_Y} \quad (17)$$

where,  $n$  is number of emotional categories,  $\bar{X}$ ,  $\bar{Y}$  and  $S_X$ ,  $S_Y$  are the mean and standard deviation values of  $X$  and  $Y$  respectively.  $corr(X, Y) = -1$  indicates that the two vectors are completely dissimilar, and  $corr(X, Y) = 1$  indicates that the vectors have perfect similarity.

The approach makes use of a thresholding mechanism to estimate the proper correlation value to find sentimentally similar words. For this, as in Mohtarami et al. (2013) we utilized the antonyms of the words. We consider two words,

#### Input:

$SAQ$ : The adjective in the question of given IQAP.  
 $SAA$ : The adjective in the answer of given IQAP.

**Output:** answer  $\in \{yes, no, uncertain\}$

#### Algorithm:

1. if  $SAQ$  or  $SAA$  are missing from our corpus then
2.     answer=*Uncertain*;
3. else if  $SS(SAQ, SAA) < 0$  then
4.     answer=*No*;
5.     else if  $SS(SAQ, SAA) > 0$  then
6.         answer=*yes*;

Figure 4. Sentiment similarity for IQAP inference

$w_i$  and  $w_j$  as similar in sentiment iff they satisfy both of the following conditions:

1.  $corr(w_i, w_j) > corr(w_i, \sim w_j)$ , and
2.  $corr(w_i, w_j) > corr(\sim w_i, w_j)$

where,  $\sim w_i$  is antonym of  $w_i$ , and  $corr(w_i, w_j)$  is obtained from Equation (17). Finally, we compute the sentiment similarity ( $SS$ ) as follows:

$$SS(w_i, w_j) = corr(w_i, w_j) - Max\{corr(w_i, \sim w_j), corr(\sim w_i, w_j)\} \quad (18)$$

Equation (18) enforces two sentimentally similar words to have weak correlation to the antonym of each others. A positive value of  $SS(...)$  indicates the words are sentimentally similar and a negative value shows that they are dissimilar.

## 5 Applications

We explain our approach in utilizing sentiment similarity between words to perform IQAP inference and SO prediction tasks respectively.

In IQAPs, we employ the sentiment similarity between the adjectives in questions and answers to interpret the indirect answers. Figure 4 shows the algorithm for this purpose.  $SS(...)$  indicates sentiment similarity computed by Equation (18). A positive  $SS$  means the words are sentimentally similar and thus the answer is *yes*. However, negative  $SS$  leads to a *no* response.

In SO-prediction task, we aim to compute more accurate SO using our sentiment similarity method. Turney and Littman (2003) proposed a method in which the SO of a word is calculated based on its semantic similarity with seven positive words minus its similarity with seven negative words as shown in Figure 5. As the similarity function,  $A(...)$ , they employed point-wise mutual information (PMI) to compute the similarity between the words. Here, we utilize the same approach, but instead of PMI we use our  $SS(...)$  measure as the similarity function.

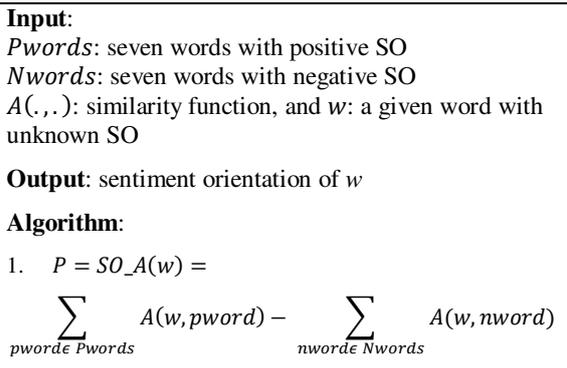


Figure 5. SO based on the similarity function *A*(.,.)

## 6 Evaluation and Results

### 6.1 Data and Settings

We used the review dataset employed by Maas et al. (2011) as the development dataset that contains movie reviews with star rating from one star (most negative) to 10 stars (most positive). We exclude the ratings 5 and 6 that are more neutral. We used this dataset to compute all the input matrices in Table 2 as well as the enriched matrix. The development dataset contains 50k movie reviews and 90k vocabulary.

We also used two datasets for the evaluation purpose: the MPQA (Wilson et al., 2005) and IQAPs (Marneffe et al., 2010) datasets. The MPQA dataset is used for SO prediction experiments, while the IQAP dataset is used for the IQAP experiments. We ignored the neutral words in MPQA dataset and used the remaining 4k opinion words. Also, the IQAPs dataset (Marneffe et al., 2010) contains 125 IQAPs and their corresponding *yes* or *no* labels as the ground truth.

### 6.2 Experimental Results

To evaluate our PSSS model, we perform experiments on the SO prediction and IQAPs inference tasks. Here, we consider six emotions for both bridged and series models. We study the effect of emotion numbers in Section 7.1. Also, we set a threshold of 0.3 for the confidence value in Equation (16), i.e. we set the confidence values smaller than the threshold to 0. We explain the effect of this parameter in Section 7.3.

#### Evaluation of SO Prediction

We evaluate the performance of our PSSS models in the SO prediction task using the algorithm explained in Figure 5 by setting our PSSS as similarity function (*A*). The results on SO prediction are presented in Table 3. The first and se-

Method	Precision	Recall	F1
PMI	56.20	56.36	55.01
ER	65.68	65.68	63.27
PSSS-SHEM	68.51	69.19	67.96
PSSS-BHEM	<b>69.39</b>	<b>70.07</b>	<b>68.68</b>

Table 3. Performance on SO prediction task

cond rows present the results of our baselines, PMI (Turney and Littman, 2003) and Expected Rating (ER) (Potts, 2011) of words respectively.

PMI extracts the semantic similarity between words using their co-occurrences. As Table 3 shows, it leads to poor performance. This is mainly due to the relatively small size of the development dataset which affects the quality of the co-occurrence information used by the PMI.

ER computes the expected rating of a word based on the distribution of the word across rating categories. The value of ER indicates the SO of the word. As shown in the two last rows of the table, the results of PSSS approach are higher than PMI and ER. The reason is that PSSS is based on the combination between sentiment space (through using ratings, and matrices  $W \times R$  in BHEM,  $D \times R$  in SHEM) and semantic space (through the input  $W \times D$  in SHEM and enriched matrix  $W \times W$  in both hidden models). However, the PMI employs only the semantic space (i.e., the co-occurrence of the words) and ER uses occurrence of the words in rating categories.

Furthermore, the PSSS model achieves higher performance with BHEM rather than SHEM. This is because the emotional vectors of the words are directly computed from the EM steps of BHEM. However, the emotional vectors of SHEM are computed after finishing the EM steps using Equation (14). This causes the SHEM model to estimate the number and type of the hidden emotions with a lower performance as compared to BHEM, although the performances of SHEM and BHEM are comparable as explained in Section 7.1.

#### Evaluation of IQAPs Inference

To apply our PSSS on IQAPs inference task, we use it as the sentiment similarity measure in the algorithm explained in Figure 4. The results are presented in Table 4. The first and second rows are baselines. The first row is the result obtained by Marneffe et al. (2010) approach. This approach is based on the similarity between the SO of the adjectives in question and answer. The second row of Table 4 show the results of using a popular semantic similarity measure, PMI, as the sentiment similarity (SS) measure in Figure 4.

Method	Prec.	Rec.	F1
Marneffe et al. (2010)	60.00	60.00	60.00
PMI	60.61	58.70	59.64
PSSS-SHEM	62.55	61.75	61.71
PSSS-BHEM (w/o WSD)	65.90	66.11	63.74
SS-BHEM (with WSD)	<b>66.95</b>	<b>67.15</b>	<b>65.66</b>

Table 4. Performance on IQAP inference task

The result shows that PMI is less effective to capture the sentiment similarity.

Our PSSS approach directly infers *yes* or *no* responses using SS between the adjectives and does not require computing SO of the adjectives. In Table 4, *PSSS-SHEM* and *PSSS-BHEM* indicate the results when we use our PSSS with SHEM and BHEM respectively. Table 4 shows the effectiveness of our sentiment similarity measure. Both models improve the performance over the baselines, while the bridged model leads to higher performance than the series model.

Furthermore, we employ Word Sense Disambiguation (WSD) to disambiguate the adjectives in the question and its corresponding answer. For example, **Q:** ... *Is that true?* **A:** *This is extraordinary and preposterous.* In the answer, the correct sense of the *extraordinary* is *unusual* and as such answer *no* can be correctly inferred. In the table, (*w/o WSD*) is based on the first sense (most common sense) of the words, whereas (*with WSD*) utilizes the real sense of the words. As Table 4 shows, WSD increases the performance. WSD could have higher effect, if more IQAPs contain adjectives with senses different from the first sense.

## 7 Analysis and Discussions

### 7.1 Number and Types of Emotions

In our PSSS approach, there is no limitation on the number and types of emotions as we assumed emotions are hidden. In this Section, we perform experiments to predict the number and type of hidden emotions.

Figure 6 and 7 show the results of the hidden models (SHEM and BHEM) on SO prediction and IQAPs inference tasks respectively with different number of emotions. As the Figures show, in both tasks, SHEM achieved high performances with 11 emotions. However, BHEM achieved high performances with six emotions. Now, the question is which emotion number should be considered? To answer this question, we further study the results as follows.

First, for SHEM, there is no significant difference between the performances with six and 11 emotions in the SO prediction task. This is the

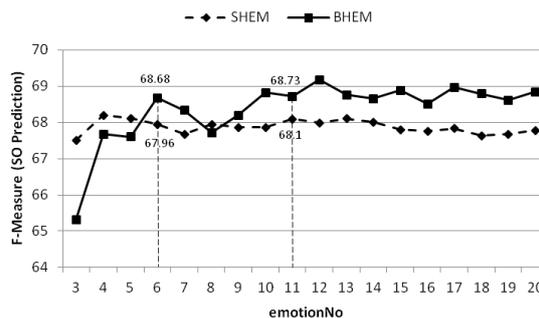


Figure 6. Performance of BHEM and SHEM on SO prediction through different #of emotions

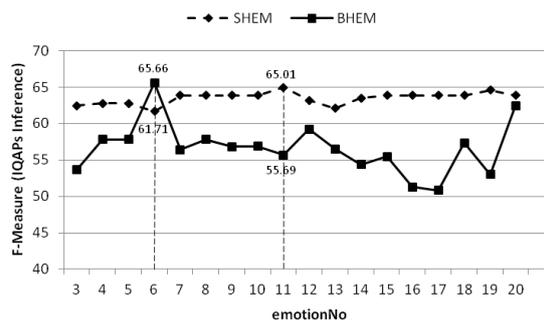


Figure 7. Performance of BHEM and SHEM on IQAPs inference through different #of emotions

same for BHEM. Also, the performances of SHEM on the IQAP inference task with six and 11 emotions are comparable. However, there is a significant difference between the performances of BHEM in six and 11 emotions. So, we consider the dimension in which both hidden emotional models present a reasonable performance over both tasks. This dimension is six here.

Second, as shown in the Figures 6 and 7, in contrast to BHEM, the performance of SHEM does not considerably change with different number of emotions over both tasks. This is because, in SHEM, the emotional vectors of the words are derived from the emotional vectors of the documents after the EM steps, see Equation (14). However, in BHEM, the emotional vectors are directly obtained from the EM steps. Thus, the bridged model is more sensitive than series model to the number of emotions. This could indicate that the bridged model is more accurate than the series model to estimate the number of emotions.

Therefore, based on the above discussion, the estimated number of emotions is six in our development dataset. This number may vary using different development datasets.

In addition to the number of emotions, their types can also be interpreted using our approach. To achieve this aim, we sort the words based on their probability values,  $P(w|e)$ , with respect to

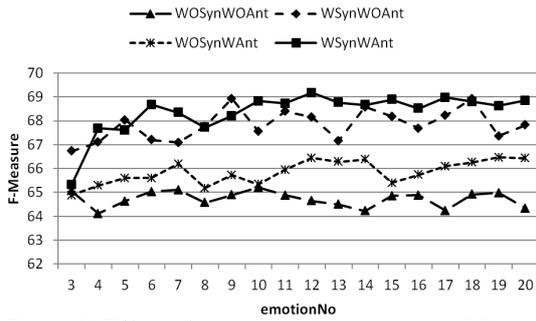


Figure 8. Effect of synonyms & antonyms in SO prediction task with different emotion numbers in BHEM

Emotion#1	Emotion#2	Emotion#3
excellent (1)	unimpressive (1)	disreputable (1)
magnificently(1)	humorlessly (1)	villian (1)
blessed (1)	paltry (1)	onslaught (1)
sublime (1)	humiliating (1)	ugly (1)
affirmation (1)	uncreative (1)	old (1)
tremendous (2)	lackluster (1)	disrupt (1)

Table 5. Sample words in three emotions

each emotion. Then, the type of the emotions can be interpreted by observing the top  $k$  words in each emotion. For example, Table 5 shows the top 6 words for three out of six emotions obtained for BHEM. The numbers in parentheses show the sense of the words. The corresponding emotions for these categories can be interpreted as "wonderful", "boring" and "disreputable", respectively.

We also observed that, in SHEM with eleven emotion numbers, some of the emotion categories have similar top  $k$  words such that they can be merged to represent the same emotion. Thus, it indicates that the BHEM is better than SHEM to estimates the number of emotions than SHEM.

## 7.2 Effect of Synsets and Antonyms

We show the important effect of synsets and antonyms in computing the sentiment similarity of words. For this purpose, we repeat the experiment for SO prediction by computing sentiment similarity of word pairs with and without using synonyms and antonyms. Figure 8 shows the results of obtained from BHEM. As the Figure shown, the highest performance can be achieved when synonyms and antonyms are used, while the lowest performance is obtained without using them. Note that, when the synonyms are not used, the entries of enriched matrix are computed using  $P(w_i|w_j)$  instead of  $P(\text{syn}(w_i)|\text{syn}(w_j))$  in the Equation (15). Also, when the antonyms are not used, the  $\text{Max}(\cdot)$  in Equation (18) is 0 and SS is computed using only correlation between words.

The results show that synonyms can improve the performance. As Figure 8 shows, the two

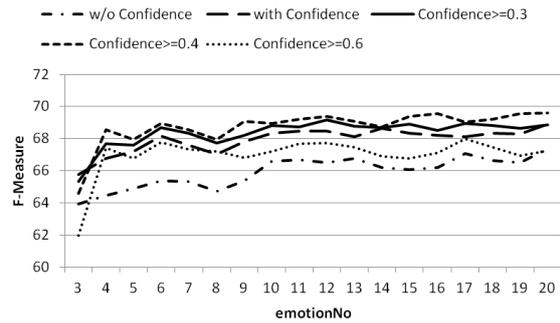


Figure 9. Effect of confidence values in SO prediction with different emotion numbers in BHEM

highest performances are obtained when we use synonyms and the two lowest performances are achieved when we don't use synonyms. This is indicates that the synsets of the words can improve the quality of the enriched matrix. The results also show that the antonyms can improve the result (compare WOSynWAnt with WOSynWOAnt). However, synonyms lead to greater improvement than antonyms (compare WSynWOAnt with WOSynWAnt).

## 7.3 Effect of Confidence Value

In Section 3.1, we defined a confidence value for each word to improve the quality of the enriched matrix. To illustrate the utility of the confidence value, we repeat the experiment for SO prediction by BHEM using all the words appears in enriched matrix with different confidence thresholds. The results are shown in Figure 9, "w/o confidence" shows the results when we don't use the confidence values, while "with confidence" shows the results when use the confidence values. Also, "confidence>x" indicates the results when we set all the confidence value smaller than x to 0. The thresholding helps to eliminate the effect of low confident words.

As Figure 9 shows, "w/o confidence" leads to the lowest performance, while "with confidence" improves the performance with different number of emotions. The thresholding is also effective. For example, a threshold like 0.3 or 0.4 improves the performance. However, if a large value (e.g., 0.6) is selected as threshold, the performance decreases. This is because a large threshold filters a large number of words from enriched model that decreases the effect of the enriched matrix.

## 7.4 Convergence Analysis

The PSSS approach is based on the EM algorithm for the BHEM (or SHEM) presented in Table 2. This algorithm performs a predefined

number of iterations or until convergence. To study the convergence of the algorithm, we repeat our experiments for SO prediction and IQAPs inference tasks using BHEM with different number of iterations. Figure 10 shows that after the first 15 iterations the performance does not change dramatically and is nearly constant when more than 30 iterations are performed. This shows that our algorithm will converge in less than 30 iterations for BHEM. We observed the same pattern in SHEM.

### 7.5 Bridged Vs. Series Model

The bridged and series models are both based on the hidden emotions that were developed to predict the sense sentiment similarity. Although their best results on the SO prediction and IQAPs inference tasks are comparable, they have some significant differences as follows:

- BHEM is considerably faster than SHEM. The reason is that, the input matrix of BHEM (i.e.,  $W \times R$ ) is significantly smaller than the input matrix of SHEM (i.e.,  $W \times D$ ).
- In BHEM, the emotional vectors are directly computed from the EM steps. However, the emotional vector of a word in SHEM is computed using the emotional vectors of the documents containing the word. This adds noises to the emotional vectors of the words.
- BHEM gives more accurate estimation over type and number of emotions versus SHEM. The reason is explained in Section 7.1.

## 8 Related Works

Sentiment similarity has not received enough attention to date. Most previous works employed semantic similarity of word pairs to address SO prediction and IQAP inference tasks. Turney and Littman (2003) proposed to compute pair-wised mutual information (PMI) between a target word and a set of seed positive and negative words to infer the SO of the target word. They also utilized Latent Semantic Analysis (LSA) (Landauer et al., 1998) as another semantic similarity measure. However, both PMI and LSA are semantic similarity measure. Similarly, Hassan and Radev (2010) presented a graph-based method for predicting SO of words. They constructed a lexical graph where nodes are words and edges connect two words with semantic similarity obtained from Wordnet (Fellbaum 1998). They propagated the SO of a set of seeds through this graph. However, such approaches did not take into account the sentiment similarity between words.

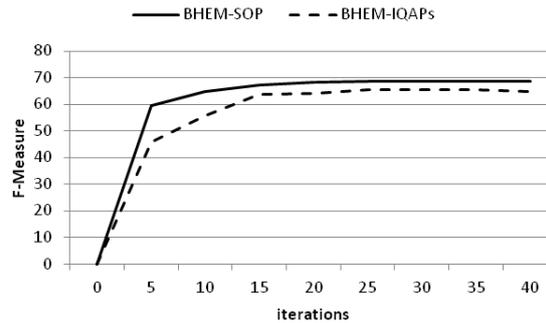


Figure 10. Convergence of BHEM

In IQAPs, Marneffe et al. (2010) inferred the *yes/no* answers using SO of the adjectives. If SO of the adjectives have different signs, then the answer conveys *no*, and Otherwise, if the absolute value of SO for the adjective in question is smaller than the absolute value of the adjective in answer, then the answer conveys *yes*, and otherwise *no*. In Mohtarami et al. (2012), we used two semantic similarity measures (PMI and LSA) for the IQAP inference task. We showed that measuring the sentiment similarities between the adjectives in question and answer leads to higher performance as compared to semantic similarity measures.

In Mohtarami et al. (2012), we proposed an approach to predict the sentiment similarity of words using their emotional vectors. We assumed that the type and number of emotions are pre-defined and our approach was based on this assumption. However, in previous research, there is little agreement about the number and types of basic emotions. Furthermore, the emotions in different dataset can be varied. We relaxed this assumption in Mohtarami et al., (2013) by considering the emotions as hidden and presented a hidden emotional model called SHEM. This paper also consider the emotions as hidden and presents another hidden emotional model called BHEM that gives more accurate estimation of the numbers and types of the hidden emotions.

## 9 Conclusion

We propose a probabilistic approach to infer the sentiment similarity between word senses with respect to automatically learned hidden emotions. We propose to utilize the correlations between reviews, ratings, and words to learn the hidden emotions. We show the effectiveness of our method in two NLP tasks. Experiments show that our sentiment similarity models lead to effective emotional vector construction and significantly outperform semantic similarity measures for the two NLP task.

## References

- Hadi Amiri and Tat S. Chua. 2012. *Mining Slang and Urban Opinion Words and Phrases from cQA Services: An Optimization Approach*. Proceedings of the fifth ACM international conference on Web search and data mining (WSDM). Pp. 193-202.
- Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.
- Ahmed Hassan and Dragomir Radev. 2010. *Identifying Text Polarity Using Random Walks*. Proceeding in the Association for Computational Linguistics (ACL). Pp: 395-403.
- Aminul Islam and Diana Inkpen. 2008. *Semantic text similarity using corpus-based word similarity and string similarity*. ACM Transactions on Knowledge Discovery from Data (TKDD).
- Carroll E. Izard. 1971. *The face of emotion*. New York: Appleton-Century-Crofts.
- Soo M. Kim and Eduard Hovy. 2004. *Determining the sentiment of opinions*. Proceeding of the Conference on Computational Linguistics (COLING). Pp: 1367-1373.
- Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. 1998. *Introduction to Latent Semantic Analysis*. Discourse Processes. Pp: 259-284.
- Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. *Learning Word Vectors for Sentiment Analysis*. Proceeding in the Association for Computational Linguistics (ACL). Pp:142-150.
- Marie-Catherine D. Marneffe, Christopher D. Manning, and Christopher Potts. 2010. *"Was it good? It was provocative." Learning the meaning of scalar adjectives*. Proceeding in the Association for Computational Linguistics (ACL). Pp: 167-176.
- Mitra Mohtarami, Hadi Amiri, Man Lan, Thanh P. Tran, and Chew L. Tan. 2012. *Sense Sentiment Similarity: An Analysis*. Proceeding of the Conference on Artificial Intelligence (AAAI).
- Mitra Mohtarami, Man Lan, and Chew L. Tan. 2013. *From Semantic to Emotional Space in Probabilistic Sense Sentiment Analysis*. Proceeding of the Conference on Artificial Intelligence (AAAI).
- Alena Neviarouskaya, Helmut Prendinger, and Mitsuru Ishizuka. 2007. *Textual Affect Sensing for Sociable and Expressive Online Communication*. Proceedings of the conference on Affective Computing and Intelligent Interaction (ACII). Pp: 218-229.
- Alena Neviarouskaya, Helmut Prendinger, and Mitsuru Ishizuka. 2009. *SentiFul: Generating a Reliable Lexicon for Sentiment Analysis*. Proceeding of the conference on Affective Computing and Intelligent Interaction (ACII). Pp: 363-368.
- Andrew Ortony and Terence J. Turner. 1990. *What's Basic About Basic Emotions*. American Psychological Association. 97(3), 315-331.
- Christopher Potts, C. 2011. *On the negativity of negation*. In Nan Li and David Lutz, eds., Proceedings of Semantics and Linguistic Theory 20, 636-659.
- Peter D. Turney and Michael L. Littman. 2003. *Measuring Praise and Criticism: Inference of Semantic Orientation from Association*. ACM Transactions on Information Systems, 21(4), 315-346.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. *Recognizing contextual polarity in phrase-level sentiment analysis*. Proceeding in HLT-EMNLP. Pp: 347-354.

# A user-centric model of voting intention from Social Media

Vasileios Lampos, Daniel Preotiuc-Pietro and Trevor Cohn

Computer Science Department

University of Sheffield, UK

{v.lampos,d.preotiuc,t.cohn}@dcs.shef.ac.uk

## Abstract

Social Media contain a multitude of user opinions which can be used to predict real-world phenomena in many domains including politics, finance and health. Most existing methods treat these problems as linear regression, learning to relate word frequencies and other simple features to a known response variable (*e.g.*, voting intention polls or financial indicators). These techniques require very careful filtering of the input texts, as most Social Media posts are irrelevant to the task. In this paper, we present a novel approach which performs high quality filtering automatically, through modelling not just words but also users, framed as a bilinear model with a sparse regulariser. We also consider the problem of modelling groups of related output variables, using a structured multi-task regularisation method. Our experiments on voting intention prediction demonstrate strong performance over large-scale input from Twitter on two distinct case studies, outperforming competitive baselines.

## 1 Introduction

Web Social Media platforms have ushered a new era in human interaction and communication. The main by-product of this activity is vast amounts of user-generated content, a type of information that has already attracted the interest of both marketers and scientists because it offers – for the first time at a large-scale – unmediated access to people’s observations and opinions.

One exciting avenue of research concentrates on mining interesting signals automatically from this stream of text input. For example, by exploiting Twitter posts, it is possible to infer time series

that correlate with financial indicators (Bollen et al., 2011), track infectious diseases (Lampos and Cristianini, 2010; Lampos et al., 2010; Paul and Dredze, 2011) and, in general, nowcast the magnitude of events emerging in real-life (Sakaki et al., 2010; Lampos and Cristianini, 2012). Other studies suggest ways for modelling opinions encapsulated in this content in order to forge branding strategies (Jansen et al., 2009) or understand various socio-political trends (Tumasjan et al., 2010; O’Connor et al., 2010; Lansdall-Welfare et al., 2012). The main theme of the aforementioned works is linear regression between word frequencies and a real-world quantity. They also tend to incorporate hand-crafted lists of search terms to filter irrelevant content and use sentiment analysis lexicons for extracting opinion bias. Consequently, they are quite often restricted to a specific application and therefore, generalise poorly to new data sets (Gayo-Avello et al., 2011).

In this paper, we propose a generic method that aims to be independent of the characteristics described above (use of search terms or sentiment analysis tools). Our approach is able to explore not only word frequencies, but also the space of users by introducing a **bilinear** formulation for this learning task. Regularised regression on both spaces allows for an automatic selection of the most important terms and users, performing at the same time an improved noise filtering. In addition, more advanced regularisation functions enable **multi-task learning** schemes that can exploit shared structure in the feature space. The latter property becomes very useful in **multi-output regression** scenarios, where selected features are expected to have correlated as well as anti-correlated impact on each output (*e.g.*, when inferring voting intentions for competing political parties).

We evaluate our methods on the domain of politics using data from the microblogging service of Twitter to infer voting trends. Our pro-

posed framework is able to successfully predict voting intentions for the top-3 and top-4 parties in the United Kingdom (UK) and Austria respectively. In both case studies – bound by different characteristics (including language, time-span and number of users) – the average prediction error is smaller than 1.5% for our best model using multi-task learning. Finally, our qualitative analysis shows that the models uncover interesting and semantically interpretable insights from the data.

## 2 Data

For the evaluation of the proposed methodologies we have created two data sets of Social Media content with different characteristics based in the UK and Austria respectively. They are used for performing regression aiming to infer voting intention polls in those countries. Data processing is performed using the TrendMiner architecture for Social Media analysis (Preoțiuc-Pietro et al., 2012).

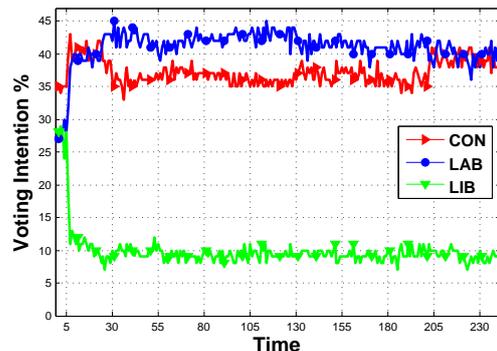
### 2.1 Tweets from users in the UK

The first data set (we refer to it as  $C_{uk}$ ) used in our experimental process consists of approx. 60 million tweets produced by approx. 42K UK Twitter users from 30/04/2010 to 13/02/2012. We assumed each user to be from the UK, if the location field in their profile matched with a list of common UK locations and their time zone was set to G.M.T. In this way, we were able to extract hundreds of thousands of UK users, from which we sub-sampled 42K users to be distributed across the UK geographical regions proportionally to their population figures.<sup>1</sup>

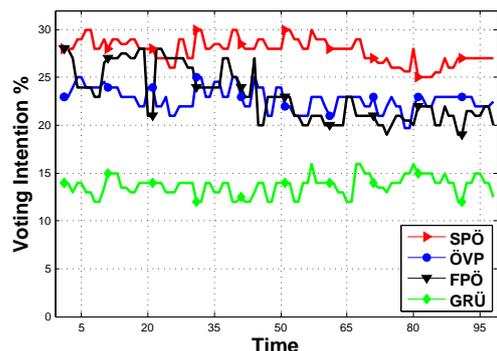
### 2.2 Tweets for Austria

The second data set ( $C_{au}$ ) is shorter in terms of the number of users involved (1.1K), its time span (25/01 to 01/12/2012) and, consequently, of the total number of tweets considered (800K). However, this time the selection of users has been made by Austrian political experts who decided which accounts to monitor by subjectively assessing the value of information they may provide towards political-oriented topics. Still, we assume that the different users will produce information of varying quality, and some should be eliminated entirely. However, we emphasise that there may be smaller

<sup>1</sup>Data collection was performed using Twitter API, <http://dev.twitter.com/>, to extract all posts for our target users.



(a) 240 voting intention polls for the 3 major parties in the UK (April 2010 to February 2012)



(b) 98 voting intention polls for the 4 major parties in Austria (January to December 2012)

Figure 1: Voting intention polls for the UK and Austria.

potential gains from user modelling compared to the UK case study. Another important distinction is language, which for this data set is primarily German with some English.

### 2.3 Ground Truth

The ground truth for training and evaluating our regression models is formed by voting intention polls from YouGov (UK) and a collection of Austrian pollsters<sup>2</sup> – as none performed high frequency polling – for the Austrian case study. We focused on the three major parties in the UK, namely Conservatives (CON), Labour (LAB) and Liberal Democrats (LBD) and the four major parties in Austria, namely the Social Democratic Party (SPÖ), People’s Party (ÖVP), Freedom Party (FPÖ) and the Green Alternative Party (GRÜ). Matching with the time spans of the data sets described in the previous sections, we have acquired 240 unique polls for the UK and 65 polls for Austria. The latter have been expanded to 98 polls by replicating the poll of day  $i$  for day

<sup>2</sup>Wikipedia, [http://de.wikipedia.org/wiki/Nationalratswahl\\_in\\_%D6sterreich\\_2013](http://de.wikipedia.org/wiki/Nationalratswahl_in_%D6sterreich_2013).

$i - 1$  where possible.<sup>3</sup> There exists some interesting variability towards the end for the UK polls (Fig. 1a), whereas for the Austrian case, the main changing point is between the second and the third party (Fig. 1b).

### 3 Methods

The textual content posted on Social Media platforms unarguably contains valuable information, but quite often it is hidden under vast amounts of unstructured user generated input. In this section, we propose a set of methods that build on one another, which aim to filter the non desirable noise and extract the most informative features not only based on word frequencies, but also by incorporating users in this process.

#### 3.1 The bilinear model

There exist a number of different possibilities for incorporating user information into a regression model. A simple approach is to expand the feature set, such that each user’s effect on the response variable can be modelled separately. Although flexible, this approach would be doomed to failure due to the sheer size of the resulting feature set, and the propensity to overfit all but the largest of training sets. One solution is to group users into different types, such as journalist, politician, activist, etc., but this presupposes a method for classification or clustering of users which is a non-trivial undertaking. Besides, these naïve approaches fail to account for the fact that most users use similar words to express their opinions, by separately parameterising the model for different users or user groups.

We propose to account for individual users while restricting all users to share the same vocabulary. This is formulated as a bilinear predictive model,

$$f(X) = \mathbf{u}^T X \mathbf{w} + \beta, \quad (1)$$

where  $X$  is an  $m \times p$  matrix of user-word frequencies and  $\mathbf{u}$  and  $\mathbf{w}$  are the model parameters. Let  $\mathcal{Q} \in \mathbb{R}^{n \times m \times p}$  be a tensor which captures our training inputs, where  $n$ ,  $m$  and  $p$  denote the considered number of samples (each sample usually refers to a day), terms and users respectively;  $\mathcal{Q}$  can simply be interpreted as  $n$  versions of  $X$  (denoted by  $\mathcal{Q}_i$  in the remainder of the script), a different one for each day, put together. Each element

<sup>3</sup>This has been carried out to ensure an adequate number of training points in the experimental process.

$\mathcal{Q}_{ijk}$  holds the frequency of term  $j$  for user  $k$  during the day  $i$  in our sample. If a user  $k$  has posted  $c_{i,k}$  tweets during day  $i$ , and  $c_{ijk} \leq c_{i,k}$  of them contain a term  $j$ , then the frequency of  $j$  for this day and user is defined as  $\mathcal{Q}_{ijk} = \frac{c_{ijk}}{c_{i,k}}$ .

Aiming to learn sparse sets of users and terms that are representative of the voting intention signal, we formulate our optimisation task as follows:

$$\{\mathbf{w}^*, \mathbf{u}^*, \beta^*\} = \underset{\mathbf{w}, \mathbf{u}, \beta}{\operatorname{argmin}} \sum_{i=1}^n (\mathbf{u}^T \mathcal{Q}_i \mathbf{w} + \beta - y_i)^2 + \psi(\mathbf{w}, \rho_1) + \psi(\mathbf{u}, \rho_2), \quad (2)$$

where  $\mathbf{y} \in \mathbb{R}^n$  is the response variable (voting intention),  $\mathbf{w} \in \mathbb{R}^m$  and  $\mathbf{u} \in \mathbb{R}^p$  denote the term and user weights respectively,  $\mathbf{u}^T \mathcal{Q}_i \mathbf{w}$  expresses the bilinear term,  $\beta \in \mathbb{R}$  is a bias term and  $\psi(\cdot)$  is a regularisation function with parameters  $\rho_1$  or  $\rho_2$ . The first term in Eq. 2 is the standard regularisation loss function, namely the sum squared error over the training instances.<sup>4</sup>

In the main formulation of our bilinear model, as the regularisation function  $\psi(\cdot)$  we use the **elastic net** (Zou and Hastie, 2005), an extension of the well-studied  $\ell_1$ -norm regulariser, known as the LASSO (Tibshirani, 1996). The  $\ell_1$ -norm regularisation has found many applications in several scientific fields as it encourages sparse solutions which reduce the possibility of overfitting and enhance the interpretability of the inferred model (Hastie et al., 2009). The elastic net applies an extra penalty on the  $\ell_2$ -norm of the weight vector, and can resolve instability issues of LASSO which arise when correlated predictors exist in the input data (Zhao and Yu, 2006). Its regularisation function  $\psi_{\text{el}}(\cdot)$  is defined by:

$$\psi_{\text{el}}(\mathbf{w}, \lambda, \alpha) = \lambda \left( \frac{1 - \alpha}{2} \|\mathbf{w}\|_2^2 + \alpha \|\mathbf{w}\|_1 \right), \quad (3)$$

where  $\lambda > 0$  and  $\alpha \in [0, 1)$ ; setting parameter  $\alpha$  to its extremes transforms elastic net to ridge regression ( $\alpha = 0$ ) or vanilla LASSO ( $\alpha = 1$ ).

Eq. 2 can be treated as a biconvex learning task (Al-Khayyal and Falk, 1983), by observing that for a fixed  $\mathbf{w}$ , learning  $\mathbf{u}$  is a convex problem and vice versa. Biconvex functions and possible applications have been well studied in the optimisation literature (Quesada and Grossmann, 1995;

<sup>4</sup>Note that other loss functions could be used here, such as logistic loss for classification, or more generally bilinear variations of Generalised Linear Models (Nelder and Wedderburn, 1972).

Pirsiavash et al., 2009). Their main advantage is the ability to solve efficiently non-convex problems by a repeated application of two convex processes, *i.e.*, a form of coordinate ascent. In our case, the bilinear technique makes it possible to explore both word and user spaces, while maintaining a modest training complexity.

Therefore, in our bilinear approach we divide learning in two phases, where we learn word and user weights respectively. For the first phase we produce the term-scores matrix  $\mathcal{V} \in \mathbb{R}^{n \times m}$  with elements given by:

$$\mathcal{V}_{ij} = \sum_{z=1}^p u_z \mathcal{Q}_{ijz}. \quad (4)$$

$\mathcal{V}$  contains weighted sums of term frequencies over all users for the considered set of days. The weights are held in  $\mathbf{u}$  and are representative of each user. The initial optimisation task is formulated as:

$$\{\mathbf{w}^*, \beta^*\} = \underset{\mathbf{w}, \beta}{\operatorname{argmin}} \|\mathcal{V}\mathbf{w} + \beta - \mathbf{y}\|_2^2 + \psi_{\text{el}}(\mathbf{w}, \lambda_1, \alpha_1), \quad (5)$$

where we aim to learn a sparse but consistent set of weights  $w^*$  for the terms of our vocabulary.

In the second phase, we are using  $\mathbf{w}^*$  to form the user-scores matrix  $\mathcal{D} \in \mathbb{R}^{n \times p}$ :

$$\mathcal{D}_{ik} = \sum_{z=1}^m w_z^* \mathcal{Q}_{izk}, \quad (6)$$

which now contains weighted sums over all terms for the same set of days. The optimisation task becomes:

$$\{\mathbf{u}^*, \beta^*\} = \underset{\mathbf{u}, \beta}{\operatorname{argmin}} \|\mathcal{D}\mathbf{u} + \beta - \mathbf{y}\|_2^2 + \psi_{\text{el}}(\mathbf{u}, \lambda_2, \alpha_2). \quad (7)$$

This process continues iteratively by inserting the weights of the second phase back to phase one, and so on until convergence. We cannot claim that a global optimum will be reached, but biconvexity guarantees that our global objective (Eq. 2) will decrease in each step of this iterative process. In the remainder of this paper, we refer to the method described above as Bilinear Elastic Net (**BEN**).

### 3.2 Exploiting term-target or user-target relationships

The previous model assumes that the response variable  $\mathbf{y}$  holds information about a single infer-

ence target. However, the task that we are addressing in this paper usually implies the existence of several targets, *i.e.*, different political parties or politicians. An important property, therefore, is the ability to perform multiple output regression. A simple way of adapting the model to the multiple output scenario is by framing a separate learning problem for each output, but tying together some of the parameters. Here we consider tying together the user weights  $\mathbf{u}$ , to enforce that the same set of users are relevant to all tasks, while learning different term weights. Note that the converse situation, where  $\mathbf{w}$ 's are tied and  $\mathbf{u}$ 's are independent, can be formulated in an equivalent manner.

Suppose that our target variable  $\mathbf{y} \in \mathbb{R}^{\tau n}$  refers now to  $\tau$  political entities,  $\mathbf{y} = [\mathbf{y}_1^T \mathbf{y}_2^T \dots \mathbf{y}_\tau^T]^T$ ; in this formation the top  $n$  elements of  $\mathbf{y}$  match to the first political entity, the next  $n$  elements to the second and so on. In the first phase of the bilinear model, we would have to solve the following optimisation task:

$$\{\mathbf{w}^*, \beta^*\} = \underset{\mathbf{w}, \beta}{\operatorname{argmin}} \sum_{i=1}^{\tau} \|\mathcal{V}\mathbf{w}_i + \beta_i - \mathbf{y}_i\|_2^2 + \sum_{i=1}^{\tau} \psi_{\text{el}}(\mathbf{w}_i, \lambda_1, \alpha_1), \quad (8)$$

where  $\mathcal{V}$  is given by Eq. 4 and  $\mathbf{w}^* \in \mathbb{R}^{\tau m}$  denotes the vector of weights which can be sliced into  $\tau$  sub-vectors  $\{\mathbf{w}_1^*, \dots, \mathbf{w}_\tau^*\}$  each one representing a political entity. In the second phase, sub-vectors  $\mathbf{w}_i^*$  are used to form the input matrices  $\mathcal{D}_i$ ,  $i \in \{1, \dots, \tau\}$  with elements given by Eq. 6. The input matrix  $\mathcal{D}'$  is formed by the vertical concatenation of all  $\mathcal{D}_i$  user score matrices, *i.e.*,  $\mathcal{D}' = [\mathcal{D}_1^T \dots \mathcal{D}_\tau^T]^T$ , and the optimisation target is equivalent to the one expressed in Eq. 7. Since  $\mathcal{D}' \in \mathbb{R}^{\tau n \times p}$ , the user weight vector  $\mathbf{u}^* \in \mathbb{R}^p$  and thus, we are learning a single weight per user and not one per political party as in the previous step.

The method described above allows learning different term weights per response variable and then binds them under a shared set of user weights. As mentioned before, one could also try the opposite (*i.e.*, start by expanding the user space); both those models can also be optimised in an iterative process. However, our experiments revealed that those approaches did not improve on the performance of BEN. Still, this behaviour could be problem-specific, *i.e.*, learning different words

from a shared set of users (and the opposite) may not be a good modelling practice for the domain of politics. Nevertheless, this observation served as a motivation for the method described in the next section, where we extract a consistent set of words and users that are weighted differently among the considered political entities.

### 3.3 Multi-task learning with the $\ell_1/\ell_2$ regulariser

All previous models – even when combining all inference targets – were not able to explore relationships across the different task domains; in our case, a task domain is defined by a specific political label or party. Ideally, we would like to make a sparse selection of words and users but with a regulariser that promotes inter-task sharing of structure, so that many features may have a positive influence towards one or more parties, but negative towards the remaining one(s). It is possible to achieve this multi-task learning property by introducing a different set of regularisation constraints in the optimisation function.

We perform multi-task learning using an extension of group LASSO (Yuan and Lin, 2006), a method known as  $\ell_1/\ell_2$  regularisation (Argyriou et al., 2008; Liu et al., 2009). Group LASSO exploits a predefined group structure on the feature space and tries to achieve sparsity in the group-level, *i.e.*, it does not perform feature selection (unlike the elastic net), but group selection. The  $\ell_1/\ell_2$  regulariser extends this notion for a  $\tau$ -dimensional response variable. The global optimisation target is now formulated as:

$$\begin{aligned} \{W^*, U^*, \beta^*\} = \\ \underset{W, U, \beta}{\operatorname{argmin}} \sum_{t=1}^{\tau} \sum_{i=1}^n (\mathbf{u}_t^T Q_i \mathbf{w}_t + \beta_t - y_{ti})^2 \quad (9) \\ + \lambda_1 \sum_{j=1}^m \|W_j\|_2 + \lambda_2 \sum_{k=1}^p \|U_k\|_2, \end{aligned}$$

where the input matrix  $Q_i$  is defined in the same way as earlier,  $W = [\mathbf{w}_1 \dots \mathbf{w}_\tau]$  is the term weight matrix (each  $\mathbf{w}_t$  refers to the  $t$ -th political entity or task), equivalently  $U = [\mathbf{u}_1 \dots \mathbf{u}_\tau]$ ,  $W_j$  and  $U_j$  denote the  $j$ -th rows of weight matrices  $W$  and  $U$  respectively, and vector  $\beta \in \mathbb{R}^\tau$  holds the bias terms per task. In this optimisation process, we aim to enforce sparsity in the feature space but in a structured manner. Notice that we are now regularising the  $\ell_{2,1}$  mixed norm of  $W$  and  $U$ , which is

defined as the sum of the row  $\ell_2$ -norms for those matrices. As a result, we expect to encourage the activation of a sparse set of features (corresponding to the rows of  $W$  and  $U$ ), but with nonzero weights across the  $\tau$  tasks (Argyriou et al., 2008). Consequently, we are performing filtering (many users and words will have zero weights) and, at the same time, assign weights of different magnitude and sign on the selected features, something that suits a political opinion mining application, where pro-A often means anti-B.

Eq. 9 can be broken into two convex tasks (following the same notion as in Eqs. 5 and 7), where we individually learn  $\{W, \beta\}$  and then  $\{U, \beta\}$ ; each step of the process is a standard linear regression problem with an  $\ell_1/\ell_2$  regulariser. Again, we are able to iterate this bilinear process and in each step convexity is guaranteed. We refer to this method as Bilinear Group  $\ell_1/\ell_2$  (BGL).

## 4 Experiments

The proposed models are evaluated on  $C_{uk}$  and  $C_{au}$  which have been introduced in Section 2. We measure predictive performance, compare it to the performance of several competitive baselines, and provide a qualitative analysis of the parameters learned by the models.

### 4.1 Data preprocessing

Basic preprocessing has been applied on the vocabulary index of  $C_{uk}$  and  $C_{au}$  aiming to filter out some of the word features and partially reduce the dimensionality of the problem. Stop words and web links were removed in both sets, together with character sequences of length  $<4$  and  $<3$  for  $C_{uk}$  and  $C_{au}$  respectively.<sup>5</sup> As the vocabulary size of  $C_{uk}$  was significantly larger, for this data set we have additionally merged Twitter hashtags (*i.e.*, words starting with ‘#’) with their exact non topic word match, where possible (by dropping the ‘#’ when the word existed in the index). After performing the preprocessing routines described above, the vocabulary sizes for  $C_{uk}$  and  $C_{au}$  were set to 80,976 and 22,917 respectively.

### 4.2 Predictive accuracy

To evaluate the predictive accuracy of our methods, we have chosen to emulate a real-life scenario

<sup>5</sup>Most of the times those character sequences were not valid words. This pattern was different in each language and thus, a different filtering threshold was applied in each data set.

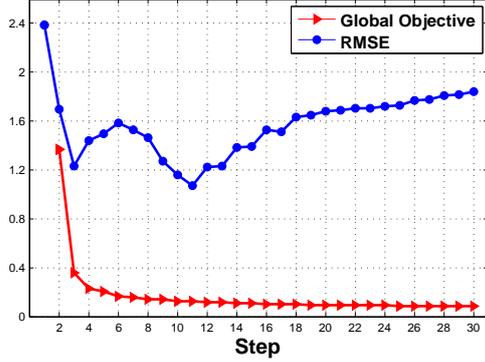


Figure 2: Global objective function and RMSE on a validation set for **BEN** in 15 iterations (30 steps) of the model.

of voting intention prediction. The evaluation process starts by using a fixed set of polls matching to consecutive time points in the past for training and validating the parameters of each model. Testing is performed on the following  $\delta$  (unseen) polls of the data set. In the next step of the evaluation process, the training/validation set is increased by merging it with the previously used test set ( $\delta$  polls), and testing is now performed on the next  $\delta$  unseen polls. In our experiments, the number of steps in this evaluation process is set to 10 and in each step the size of the test set is set to  $\delta = 5$  polls. Hence, each model is tested on 50 unseen and consecutive in time samples. The loss function in our evaluation is the standard Mean Square Error (**MSE**), but to allow a better interpretation of the results, we display its root (**RMSE**) in tables and figures.<sup>6</sup>

The parameters of each model ( $\alpha_i$  for **BEN** and  $\lambda_i$  for **BEN** and **BGL**,  $i \in \{1, 2\}$ ) are optimised using a held-out validation set by performing grid search. Note that it may be tempting to adapt the regularisation parameters in each phase of the iterative training loop, however this would change the global objective (see Eqs. 2 and 9) and thus convergence will not be guaranteed. A key question is how many iterations of training are required to reach convergence. Figure 2 illustrates how the **BEN** global objective function (Eq. 2) converges during this iterative process and the model’s performance on an unseen validation set. Notice that there is a large performance improvement after the first step (which alone is a linear solver), but overfitting occurs after step 11. Based on this result, for subsequent experiments we run the training process for two iterations (4 steps), and take the

<sup>6</sup>RMSE has the same metric units as the response variable.

	<b>CON</b>	<b>LAB</b>	<b>LBD</b>	$\mu$
<b>B<math>_{\mu}</math></b>	2.272	1.663	1.136	1.69
<b>B<math>_{last}</math></b>	2	2.074	1.095	1.723
<b>LEN</b>	3.845	2.912	2.445	3.067
<b>BEN</b>	1.939	1.644	1.136	1.573
<b>BGL</b>	<b>1.785</b>	<b>1.595</b>	<b>1.054</b>	<b>1.478</b>

Table 1: UK case study — Average RMSEs representing the error of the inferred voting intention percentage for the 10-step validation process;  $\mu$  denotes the mean RMSE across the three political parties for each baseline or inference method.

	<b>SPÖ</b>	<b>ÖVP</b>	<b>FPÖ</b>	<b>GRÜ</b>	$\mu$
<b>B<math>_{\mu}</math></b>	1.535	1.373	3.3	1.197	1.851
<b>B<math>_{last}</math></b>	<b>1.148</b>	1.556	<b>1.639</b>	1.536	1.47
<b>LEN</b>	1.291	1.286	2.039	<b>1.152</b>	1.442
<b>BEN</b>	1.392	1.31	2.89	1.205	1.699
<b>BGL</b>	1.619	<b>1.005</b>	1.757	1.374	<b>1.439</b>

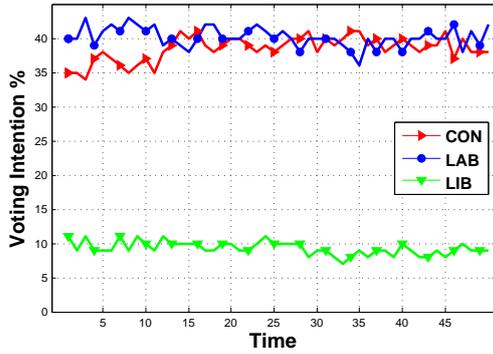
Table 2: Austrian case study — Average RMSEs for the 10-step validation process.

best performing model on the held-out validation set.

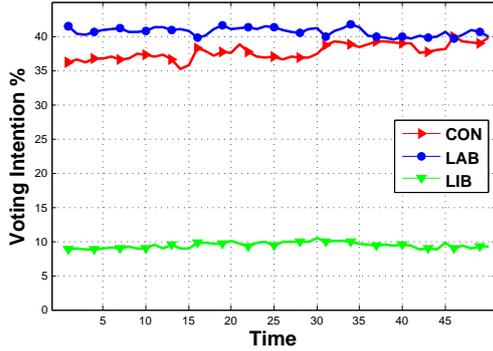
We compare the performance of our methods with three baselines. The first makes a constant prediction of the mean value of the response variable  $\mathbf{y}$  in the training set (**B $_{\mu}$** ); the second predicts the last value of  $\mathbf{y}$  (**B $_{last}$** ); and the third baseline (**LEN**) is a linear regression over the terms using elastic net regularisation. Recalling that each test set is made of 5 polls, **B $_{last}$**  should be considered as a hard baseline to beat<sup>7</sup> given that voting intentions tend to have a smooth behaviour. Moreover, improving on **LEN** partly justifies the usefulness of a bilinear approach compared to a linear one.

Performance results comparing inferred voting intention percentages and polls for  $C_{uk}$  and  $C_{au}$  are presented in Tables 1 and 2 respectively. For the UK case study, both **BEN** and **BGL** are able to beat all baselines in average performance across all parties. However in the Austrian case study, **LEN** performs better than **BEN**, something that could be justified by the fact that the users in  $C_{au}$  were selected by domain experts, and consequently there was not much gain to be had by filtering them further. Nevertheless, the difference in performance was rather small (approx. 0.26% error) and the in-

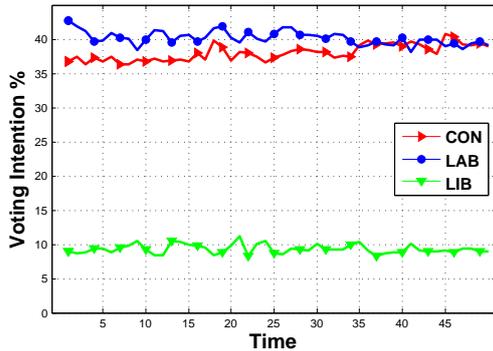
<sup>7</sup>The last response value could be easily included as a feature in the model, and would likely improve predictive performance.



(a) Ground Truth (polls)



(b) BEN



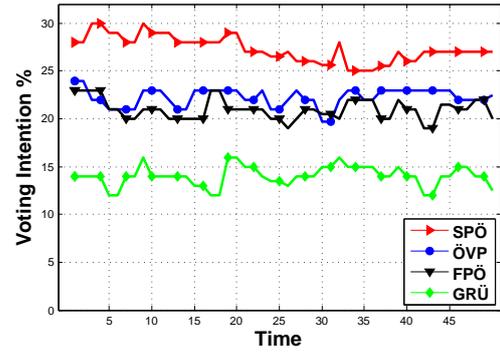
(c) BGL

Figure 3: UK case study — Voting intention inference results (50 polls, 3 parties). Sub-figure 3a is a plot of ground truth as presented in voting intention polls (Fig. 1a).

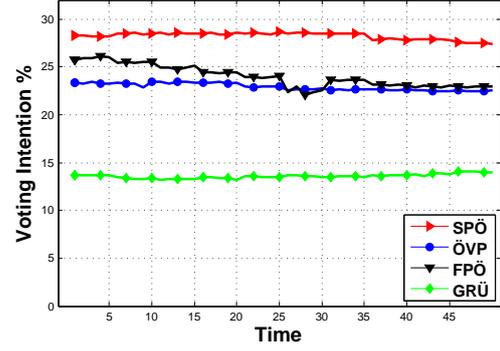
ferences of LEN and BEN followed a very similar pattern ( $\bar{\rho} = .94$  with  $p < 10^{-10}$ ).<sup>8</sup> Multi-task learning (BGL) delivered the best inference performance in both case studies, which was on average smaller than 1.48% (RMSE).

Inferences for both BEN and BGL have been plotted on Figures 3 and 4. They are presented as continuous lines of 50 inferred points (per party) which are created by concatenating the inferences

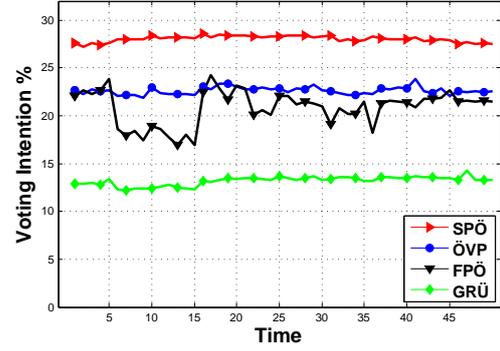
<sup>8</sup>Pearson’s linear correlation averaged across the four Austrian parties.



(a) Ground Truth (polls)



(b) BEN



(c) BGL

Figure 4: Austrian case study — Voting intention inference results (50 polls, 4 parties). Sub-figure 4a is a plot of ground truth as presented in voting intention polls (Fig. 1b).

on all test sets.<sup>9</sup> For the UK case study, one may observe that BEN (Fig. 3b) cannot register any change – with the exception of one test point – in the leading party fight (CON versus LAB); BGL (Fig. 3c) performs much better in that aspect. In the Austrian case study this characteristic becomes more obvious. BEN (Fig. 4b) consistently predicts the wrong ranking of ÖVP and FPÖ, whereas BGL (Fig. 4c) does much better. Most importantly, a

<sup>9</sup>Voting intention polls were plotted separately to allow a better presentation.

Party	Tweet	Score	Author
CON	PM in friendly chat with top EU mate, Sweden's Fredrik Reinfeldt, before family photo	1.334	Journalist
	Have Liberal Democrats broken electoral rules? Blog on Labour complaint to cabinet secretary	-0.991	Journalist
LAB	Blog Post Liverpool: City of Radicals Website now Live <link> #liverpool #art	1.954	Art Fanzine
	I am so pleased to hear Paul Savage who worked for the Labour group has been Appointed the Marketing manager for the baths hall GREAT NEWS	-0.552	Politician (Labour)
LBD	RT @user: Must be awful for TV bosses to keep getting knocked back by all the women they ask to host election night (via @user)	0.874	LibDem MP
	Blog Post Liverpool: City of Radicals 2011 – More Details Announced #liverpool #art	-0.521	Art Fanzine
SPÖ	Inflationsrate in Ö. im Juli leicht gesunken: von 2,2 auf 2,1%. Teurer wurde Wohnen, Wasser, Energie. <b>Translation:</b> <i>Inflation rate in Austria slightly down in July from 2,2 to 2,1%. Accommodation, Water, Energy more expensive.</i>	0.745	Journalist
ÖVP	Hans Rauscher zu Felix #Baumgartner "A klaner Hitler" <link> <b>Translation:</b> <i>Hans Rauscher on Felix #Baumgartner "A little Hitler" &lt;link&gt;</i>	-1.711	Journalist
	#IchPirat setze mich dafür ein, dass eine große Koalition _mathematisch_ verhindert wird! 1.Geige: #Gruene + #FPOe + #OeVP <b>Translation:</b> <i>#IPirate am committed to prevent a grand coalition mathematically! Calling the tune: #Greens + #FPO + #OVP</i>	4.953	User
FPÖ	kann das buch "res publica" von johannes #voggenhuber wirklich empfehlen! so zum nachdenken und so... #europa #demokratie <b>Translation:</b> <i>can really recommend the book "res publica" by johannes #voggenhuber! Food for thought and so on #europe #democracy</i>	-2.323	User
	Neue Kampagne der #Krone zur #Wehrpflicht: "GIB BELLO EINE STIMME!" <b>Translation:</b> <i>New campaign by the #Krone on #Conscription: "GIVE WOOFY A VOICE!"</i>	7.44	Political satire
GRÜ	Kampagne der Wiener SPÖ "zum Zusammenleben" spielt Rechtspopulisten in die Hände <link> <b>Translation:</b> <i>Campaign of the Viennese SPÖ on "Living together" plays right into the hands of right-wing populists &lt;link&gt;</i>	-3.44	Human Rights
	Protestsong gegen die Abschaffung des Bachelor-Studiums Internationale Entwicklung: <link> #IEbleibt #unibrennt #uniwut <b>Translation:</b> <i>Protest songs against the closing-down of the bachelor course of International Development: &lt;link&gt; #IDremains #uniburns #unirage</i>	1.45	Student Union
	Pilz "ich will in dieser Republik weder kriminelle Asylwerber, noch kriminelle orange Politiker" - BZÖ-Abschiebung ok, aber wohin? #amPunkt <b>Translation:</b> <i>Pilz "i want neither criminal asylum-seekers, nor criminal orange politicians in this republic" - BZÖ-Deportation OK, but where? #amPunkt</i>	-2.172	User

Table 3: Examples of tweets amongst the ones with top positive and negative scores per party for both  $C_{uk}$  and  $C_{au}$  data sets (tweets in Austrian have been translated in English as well). Notice that weight magnitude may differ per case study and party as they are based on the range of the response variable and the total number of selected features.

general observation is that BEN's predictions are smooth and do not vary significantly with time. This might be a result of overfitting the model to a single response variable which usually has a smooth behaviour. On the contrary, the multi-task learning property of BGL reduces this type of overfitting providing more statistical evidence for the terms and users and thus, yielding not only a better inference performance, but also a more accurate model.

### 4.3 Qualitative Analysis

In this section, we refer to features that have been selected and weighted as significant by our bi-

linear learning functions. Based on the weights for the word and the user spaces that we retrieve after the application of BGL in the last step of the evaluation process (see the previous section), we compute a score (weighted sum) for each tweet in our training data sets for both  $C_{uk}$  and  $C_{au}$ . Table 3 shows examples of interesting tweets amongst the top weighted ones (positively as well as negatively) per party. Together with their text (anonymised for privacy reasons) and scores, we also provide an attribute for the author (if present). In the displayed tweets for the UK study, the only possible outlier is the 'Art Fanzine'; still, it seems to register a consistent behaviour (positive towards

LAB, negative towards LBD) and, of course, hidden, indirect relationships may exist between political opinion and art. The Austrian case study revealed even more interesting tweets since training was conducted on data from a very active pre-election period (we made an effort to translate those tweets in English language as well). For a better interpretation of the presented tweets, it may be useful to know that ‘*Johannes Voggenhuber*’ (who receives a positive comment for his book) and ‘*Peter Pilz*’ (whose comment is questioned) are members of GRÜ, ‘*Krone*’ (or *Kronen Zeitung*) is the major newspaper in Austria<sup>10</sup> and that FPÖ is labelled as a far right party, something that may cause various reactions from ‘*Human Rights*’ organisations.

## 5 Related Work

The topic of political opinion mining from Social Media has been the focus of various recent research works. Several papers have presented methods that aim to predict the result of an election (Tumasjan et al., 2010; Bermingham and Smeaton, 2011) or to model voting intention and other kinds of socio-political polls (O’Connor et al., 2010; Lampos, 2012). Their common feature is a methodology based on a meta-analysis of word frequencies using off-the-shelf sentiment tools such as LIWC (Pennebaker et al., 2007) or Senti-WordNet (Esuli and Sebastiani, 2006). Moreover, the proposed techniques tend to incorporate posting volume figures as well as hand-crafted lists of words relevant to the task (*e.g.*, names of politicians or parties) in order to filter the content successfully.

Such papers have been criticised as their methods do not generalise when applied on different data sets. According to the work in (Gayo-Avello et al., 2011), the methods presented in (Tumasjan et al., 2010) and (O’Connor et al., 2010) failed to predict the result of US congressional elections in 2009. We disagree with the arguments supporting the statement “you cannot predict elections with Twitter” (Gayo-Avello, 2012), as many times in the past actual voting intention polls have also failed to predict election outcomes, but we agree that most methods that have been proposed so far were not entirely generic. It is a fact that the

<sup>10</sup>“Accused of abusing its near monopoly to manipulate public opinion in Austria”, Wikipedia, 19/02/2013, [http://en.wikipedia.org/wiki/Kronen\\_Zeitung](http://en.wikipedia.org/wiki/Kronen_Zeitung).

majority of sentiment analysis tools are English-specific (or even American English) and, most importantly, political word lists (or ontologies) change in time, per country and per party; hence, generalisable methods should make an effort to limit reliance from such tools.

Furthermore, our work – indirectly – meets the guidelines proposed in (Metaxas et al., 2011) as we have developed a framework of “well-defined” algorithms that are “Social Web aware” (since the bilinear approach aims to improve noise filtering) and that have been tested on two evaluation scenarios with distinct characteristics.

## 6 Conclusions and Future Work

We have presented a novel method for text regression that exploits both word and user spaces by solving a bilinear optimisation task, and an extension that applies multi-task learning for multi-output inference. Our approach performs feature selection – hence, noise filtering – on large-scale user-generated inputs automatically, generalises across two languages without manual adaptations and delivers some significant improvements over strong performance baselines (< 1.5% error when predicting polls). The application domain in this paper was politics, though the presented methods are generic and could be easily applied on various other domains, such as health or finance.

Future work may investigate further modelling improvements achieved by applying different regularisation functions as well as the adaptation of the presented models to classification problems. Finally, in the application level, we aim at an in-depth analysis of patterns and characteristics in the extracted sets of features by collaborating with domain experts (*e.g.*, political analysts).

## Acknowledgments

This work was funded by the TrendMiner project (EU-FP7-ICT n.287863). All authors would like to thank the political analysts (and especially Paul Ringler) from SORA<sup>11</sup> for their useful insights on politics in Austria.

<sup>11</sup>SORA – Institute for Social Research and Consulting, <http://www.sora.at>.

## References

- Faiz A Al-Khayyal and James E Falk. 1983. Jointly Constrained Biconvex Programming. *Mathematics of Operations Research*, 8(2):273–286.
- Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. 2008. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, January.
- Adam Bermingham and Alan F Smeaton. 2011. On using Twitter to monitor political sentiment and predict election results. In *Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology (SAAIP 2011)*, pages 2–10, November.
- Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8, March.
- Andrea Esuli and Fabrizio Sebastiani. 2006. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceeding of the 5th Conference on Language Resources and Evaluation (LREC)*, pages 417–422.
- Daniel Gayo-Avello, Panagiotis T Metaxas, and Eni Mustafaraj. 2011. Limits of Electoral Predictions using Twitter. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM)*, pages 490–493.
- Daniel Gayo-Avello. 2012. No, You Cannot Predict Elections with Twitter. *IEEE Internet Computing*, 16(6):91–94, November.
- Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer.
- Bernard J Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. 2009. Twitter power: Tweets as electronic word of mouth. *Journal of the American Society for Information Science and Technology*, 60(11):2169–2188.
- Vasileios Lampos and Nello Cristianini. 2010. Tracking the flu pandemic by monitoring the Social Web. In *2nd IAPR Workshop on Cognitive Information Processing*, pages 411–416. IEEE Press.
- Vasileios Lampos and Nello Cristianini. 2012. Nowcasting Events from the Social Web with Statistical Learning. *ACM Transactions on Intelligent Systems and Technology*, 3(4):1–22, September.
- Vasileios Lampos, Tijn De Bie, and Nello Cristianini. 2010. Flu Detector - Tracking Epidemics on Twitter. In *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, pages 599–602. Springer.
- Vasileios Lampos. 2012. On voting intentions inference from Twitter content: a case study on UK 2010 General Election. *CoRR*, April.
- Thomas Lansdall-Welfare, Vasileios Lampos, and Nello Cristianini. 2012. Effects of the recession on public mood in the UK. In *Proceedings of the 21st international conference companion on World Wide Web, WWW '12 Companion*, pages 1221–1226. ACM.
- Jun Liu, Shuiwang Ji, and Jieping Ye. 2009. Multi-task feature learning via efficient  $l_{2,1}$ -norm minimization. pages 339–348, June.
- Panagiotis T Metaxas, Eni Mustafaraj, and Daniel Gayo-Avello. 2011. How (Not) To Predict Elections. In *IEEE 3rd International Conference on Social Computing (SocialCom)*, pages 165–171. IEEE Press.
- John A Nelder and Robert W M Wedderburn. 1972. Generalized Linear Models. *Journal of the Royal Statistical Society - Series A (General)*, 135(3):370.
- Brendan O’Connor, Ramnath Balasubramanyan, Bryan R Routledge, and Noah A Smith. 2010. From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, pages 122–129. AAAI Press.
- Michael J Paul and Mark Dredze. 2011. You Are What You Tweet: Analyzing Twitter for Public Health. *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*, pages 265–272.
- James W Pennebaker, Cindy K Chung, Molly Ireland, Amy Gonzales, and Roger J Booth. 2007. The Development and Psychometric Properties of LIWC2007. Technical report, Universities of Texas at Austin & University of Auckland, New Zealand.
- Hamed Pirsiavash, Deva Ramanan, and Charles Fowlkes. 2009. Bilinear classifiers for visual recognition. In *Advances in Neural Information Processing Systems*, volume 22, pages 1482–1490.
- Daniel Preoțiuc-Pietro, Sina Samangooei, Trevor Cohn, Nicholas Gibbins, and Mahesan Niranjan. 2012. Trendminer: An Architecture for Real Time Analysis of Social Media Text. In *Sixth International AAAI Conference on Weblogs and Social Media*, pages 38–42. AAAI Press, July.
- Ignacio Quesada and Ignacio E Grossmann. 1995. A global optimization algorithm for linear fractional and bilinear programs. *Journal of Global Optimization*, 6(1):39–76, January.
- Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World Wide Web (WWW)*, pages 851–860. ACM.
- Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society - Series B (Methodological)*, 58(1):267–288.

- Andranik Tumasjan, Timm O Sprenger, Philipp G Sandner, and Isabell M Welp. 2010. Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, pages 178–185. AAAI.
- Ming Yuan and Yi Lin. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society - Series B: Statistical Methodology*, 68(1):49–67.
- Peng Zhao and Bin Yu. 2006. On model selection consistency of Lasso. *Journal of Machine Learning Research*, 7(11):2541–2563.
- Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, April.

# Using Supervised Bigram-based ILP for Extractive Summarization

Chen Li, Xian Qian, and Yang Liu

The University of Texas at Dallas

Computer Science Department

chenli, qx, yangl@hlt.utdallas.edu

## Abstract

In this paper, we propose a bigram based supervised method for extractive document summarization in the integer linear programming (ILP) framework. For each bigram, a regression model is used to estimate its frequency in the reference summary. The regression model uses a variety of indicative features and is trained discriminatively to minimize the distance between the estimated and the ground truth bigram frequency in the reference summary. During testing, the sentence selection problem is formulated as an ILP problem to maximize the bigram gains. We demonstrate that our system consistently outperforms the previous ILP method on different TAC data sets, and performs competitively compared to the best results in the TAC evaluations. We also conducted various analysis to show the impact of bigram selection, weight estimation, and ILP setup.

## 1 Introduction

Extractive summarization is a sentence selection problem: identifying important summary sentences from one or multiple documents. Many methods have been developed for this problem, including supervised approaches that use classifiers to predict summary sentences, graph based approaches to rank the sentences, and recent global optimization methods such as integer linear programming (ILP) and submodular methods. These global optimization methods have been shown to be quite powerful for extractive summarization, because they try to select important sentences and remove redundancy at the same time under the length constraint.

Gillick and Favre (Gillick and Favre, 2009) introduced the concept-based ILP for summariza-

tion. Their system achieved the best result in the TAC 09 summarization task based on the ROUGE evaluation metric. In this approach the goal is to maximize the sum of the weights of the language concepts that appear in the summary. They used bigrams as such language concepts. The association between the language concepts and sentences serves as the constraints. This ILP method is formally represented as below (see (Gillick and Favre, 2009) for more details):

$$\max \quad \sum_i w_i c_i \quad (1)$$

$$s.t. \quad s_j Occ_{ij} \leq c_i \quad (2)$$

$$\sum_j s_j Occ_{ij} \geq c_i \quad (3)$$

$$\sum_j l_j s_j \leq L \quad (4)$$

$$c_i \in \{0, 1\} \forall i \quad (5)$$

$$s_j \in \{0, 1\} \forall j \quad (6)$$

$c_i$  and  $s_j$  are binary variables (shown in (5) and (6)) that indicate the presence of a concept and a sentence respectively.  $w_i$  is a concept's weight and  $Occ_{ij}$  means the occurrence of concept  $i$  in sentence  $j$ . Inequalities (2)(3) associate the sentences and concepts. They ensure that selecting a sentence leads to the selection of all the concepts it contains, and selecting a concept only happens when it is present in at least one of the selected sentences.

There are two important components in this concept-based ILP: one is how to select the concepts ( $c_i$ ); the second is how to set up their weights ( $w_i$ ). Gillick and Favre (Gillick and Favre, 2009) used bigrams as concepts, which are selected from a subset of the sentences, and their document frequency as the weight in the objective function.

In this paper, we propose to find a candidate summary such that the language concepts (e.g., bigrams) in this candidate summary and the reference summary can have the same frequency. We expect this restriction is more consistent with the

ROUGE evaluation metric used for summarization (Lin, 2004). In addition, in the previous concept-based ILP method, the constraints are with respect to the appearance of language concepts, hence it cannot distinguish the importance of different language concepts in the reference summary. Our method can decide not only which language concepts to use in ILP, but also the frequency of these language concepts in the candidate summary. To estimate the bigram frequency in the summary, we propose to use a supervised regression model that is discriminatively trained using a variety of features. Our experiments on several TAC summarization data sets demonstrate this proposed method outperforms the previous ILP system and often the best performing TAC system.

## 2 Proposed Method

### 2.1 Bigram Gain Maximization by ILP

We choose bigrams as the language concepts in our proposed method since they have been successfully used in previous work. In addition, we expect that the bigram oriented ILP is consistent with the ROUGE-2 measure widely used for summarization evaluation.

We start the description of our approach for the scenario where a human abstractive summary is provided, and the task is to select sentences to form an extractive summary. Then Our goal is to make the bigram frequency in this system summary as close as possible to that in the reference. For each bigram  $b$ , we define its gain:

$$\text{Gain}(b, \text{sum}) = \min\{n_{b,ref}, n_{b,sum}\} \quad (7)$$

where  $n_{b,ref}$  is the frequency of  $b$  in the reference summary, and  $n_{b,sum}$  is the frequency of  $b$  in the automatic summary. The gain of a bigram is no more than its frequency in the reference summary, hence adding redundant bigrams will not increase the gain.

The total gain of an extractive summary is defined as the sum of every bigram gain in the summary:

$$\begin{aligned} \text{Gain}(\text{sum}) &= \sum_b \text{Gain}(b, \text{sum}) \\ &= \sum_b \min\{n_{b,ref}, \sum_s z(s) * n_{b,s}\} \end{aligned} \quad (8)$$

where  $s$  is a sentence in the document,  $n_{b,s}$  is the frequency of  $b$  in sentence  $s$ ,  $z(s)$  is a binary variable, indicating whether  $s$  is selected in the

summary. The goal is to find  $z$  that maximizes  $\text{Gain}(\text{sum})$  (formula (8)) under the length constraint  $L$ .

This problem can be casted as an ILP problem. First, using the fact that

$$\min\{a, x\} = 0.5(-|x - a| + x + a), \quad x, a \geq 0$$

we have

$$\begin{aligned} \sum_b \min\{n_{b,ref}, \sum_s z(s) * n_{b,s}\} &= \\ \sum_b 0.5 * (-|n_{b,ref} - \sum_s z(s) * n_{b,s}| + & \\ n_{b,ref} + \sum_s z(s) * n_{b,s}) & \end{aligned}$$

Now the problem is equivalent to:

$$\begin{aligned} \max_z \quad & \sum_b (-|n_{b,ref} - \sum_s z(s) * n_{b,s}| + \\ & n_{b,ref} + \sum_s z(s) * n_{b,s}) \\ \text{s.t.} \quad & \sum_s z(s) * |S| \leq L; \quad z(s) \in \{0, 1\} \end{aligned}$$

This is equivalent to the ILP:

$$\max \quad \sum_b (\sum_s z(s) * n_{b,s} - C_b) \quad (9)$$

$$\text{s.t.} \quad \sum_s z(s) * |S| \leq L \quad (10)$$

$$z(s) \in \{0, 1\} \quad (11)$$

$$-C_b \leq n_{b,ref} - \sum_s z(s) * n_{b,s} \leq C_b \quad (12)$$

where  $C_b$  is an auxiliary variable we introduce that is equal to  $|n_{b,ref} - \sum_s z(s) * n_{b,s}|$ , and  $n_{b,ref}$  is a constant that can be dropped from the objective function.

### 2.2 Regression Model for Bigram Frequency Estimation

In the previous section, we assume that  $n_{b,ref}$  is at hand (reference abstractive summary is given) and propose a bigram-based optimization framework for extractive summarization. However, for the summarization task, the bigram frequency is unknown, and thus our first goal is to estimate such frequency. We propose to use a regression model for this.

Since a bigram's frequency depends on the summary length ( $L$ ), we use a normalized frequency

in our method. Let  $n_{b,ref} = N_{b,ref} * L$ , where  $N_{b,ref} = \frac{n(b,ref)}{\sum_b n(b,ref)}$  is the normalized frequency in the summary. Now the problem is to automatically estimate  $N_{b,ref}$ .

Since the normalized frequency  $N_{b,ref}$  is a real number, we choose to use a logistic regression model to predict it:

$$N_{b,ref} = \frac{\exp\{w'f(b)\}}{\sum_j \exp\{w'f(b_j)\}} \quad (13)$$

where  $f(b_j)$  is the feature vector of bigram  $b_j$  and  $w'$  is the corresponding feature weight. Since even for identical bigrams  $b_i = b_j$ , their feature vectors may be different ( $f(b_i) \neq f(b_j)$ ) due to their different contexts, we sum up frequencies for identical bigrams  $\{b_i | b_i = b\}$ :

$$\begin{aligned} N_{b,ref} &= \sum_{i, b_i=b} N_{b_i,ref} \\ &= \frac{\sum_{i, b_i=b} \exp\{w'f(b_i)\}}{\sum_j \exp\{w'f(b_j)\}} \end{aligned} \quad (14)$$

To train this regression model using the given reference abstractive summaries, rather than trying to minimize the squared error as typically done, we propose a new objective function. Since the normalized frequency satisfies the probability constraint  $\sum_b N_{b,ref} = 1$ , we propose to use KL divergence to measure the distance between the estimated frequencies and the ground truth values. The objective function for training is thus to minimize the KL distance:

$$\min \sum_b \tilde{N}_{b,ref} \log \frac{\tilde{N}_{b,ref}}{N_{b,ref}} \quad (15)$$

where  $\tilde{N}_{b,ref}$  is the true normalized frequency of bigram  $b$  in reference summaries.

Finally, we replace  $N_{b,ref}$  in Formula (15) with Eq (14) and get the objective function below:

$$\max \sum_b \tilde{N}_{b,ref} \log \frac{\sum_{i, b_i=b} \exp\{w'f(b_i)\}}{\sum_j \exp\{w'f(b_j)\}} \quad (16)$$

This shares the same form as the contrastive estimation proposed by (Smith and Eisner, 2005). We use gradient decent method for parameter estimation, initial  $w$  is set with zero.

## 2.3 Features

Each bigram is represented using a set of features in the above regression model. We use two types of features: word level and sentence level features. Some of these features have been used in previous work (Aker and Gaizauskas, 2009; Brandow et al., 1995; Edmundson, 1969; Radev, 2001):

- Word Level:

- **1. Term frequency1:** The frequency of this bigram in the given topic.
- **2. Term frequency2:** The frequency of this bigram in the selected sentences<sup>1</sup>.
- **3. Stop word ratio:** Ratio of stop words in this bigram. The value can be  $\{0, 0.5, 1\}$ .
- **4. Similarity with topic title:** The number of common tokens in these two strings, divided by the length of the longer string.
- **5. Similarity with description of the topic:** Similarity of the bigram with topic description (see next data section about the given topics in the summarization task).

- Sentence Level: (information of sentence containing the bigram)

- **6. Sentence ratio:** Number of sentences that include this bigram, divided by the total number of the selected sentences.
- **7. Sentence similarity:** Sentence similarity with topic's query, which is the concatenation of topic title and description.
- **8. Sentence position:** Sentence position in the document.
- **9. Sentence length:** The number of words in the sentence.
- **10. Paragraph starter:** Binary feature indicating whether this sentence is the beginning of a paragraph.

## 3 Experiments

### 3.1 Data

We evaluate our method using several recent TAC data sets, from 2008 to 2011. The TAC summarization task is to generate at most 100 words summaries from 10 documents for a given topic query (with a title and more detailed description). For model training, we also included two years' DUC data (2006 and 2007). When evaluating on one TAC data set, we use the other years of the TAC data plus the two DUC data sets as the training data.

<sup>1</sup>See next section about the sentence selection step

### 3.2 Summarization System

We use the same system pipeline described in (Gillick et al., 2008; McDonald, 2007). The key modules in the ICSI ILP system (Gillick et al., 2008) are briefly described below.

- Step 1: Clean documents, split text into sentences.
- Step 2: Extract bigrams from all the sentences, then select those bigrams with document frequency equal to more than 3. We call this subset as initial bigram set in the following.
- Step 3: Select relevant sentences that contain at least one bigram from the initial bigram set.
- Step 4: Feed the ILP with sentences and the bigram set to get the result.
- Step 5: Order sentences identified by ILP as the final result of summary.

The difference between the ICSI and our system is in the 4th step. In our method, we first extract all the bigrams from the selected sentences and then estimate each bigram’s  $N_{b,ref}$  using the regression model. Then we use the top-n bigrams with their  $N_{b,ref}$  and all the selected sentences in our proposed ILP module for summary sentence selection. When training our bigram regression model, we use each of the 4 reference summaries separately, i.e., the bigram frequency is obtained from one reference summary. The same pre-selection of sentences described above is also applied in training, that is, the bigram instances used in training are from these selected sentences and the reference summary.

## 4 Experiment and Analysis

### 4.1 Experimental Results

Table 1 shows the ROUGE-2 results of our proposed system, the ICSI system, and also the best performing system in the NIST TAC evaluation. We can see that our proposed system consistently outperforms ICSI ILP system (the gain is statistically significant based on ROUGE’s 95% confidence interval results). Compared to the best reported TAC result, our method has better performance on three data sets, except 2011 data. Note that the best performing system for the 2009 data is the ICSI ILP system, with an additional compression step. Our ILP method is purely extrac-

tive. Even without using compression, our approach performs better than the full ICSI system. The best performing system for the 2011 data also has some compression module. We expect that after applying sentence compression and merging, we will have even better performance, however, our focus in this paper is on the bigram-based extractive summarization.

	ICSI ILP	Proposed System	TAC Rank1 System
2008	0.1023	<b>0.1076</b>	0.1038
2009	0.1160	<b>0.1246</b>	0.1216
2010	0.1003	<b>0.1067</b>	0.0957
2011	0.1271	0.1327	<b>0.1344</b>

Table 1: ROUGE-2 summarization results.

There are several differences between the ICSI system and our proposed method. First is the bigrams (concepts) used. We use the top 100 bigrams from our bigram estimation module; whereas the ICSI system just used the initial bigram set described in Section 3.2. Second, the weights for those bigrams differ. We used the estimated value from the regression model; the ICSI system just uses the bigram’s document frequency in the original text as weight. Finally, two systems use different ILP setups. To analyze which factors (or all of them) explain the performance difference, we conducted various controlled experiments for these three factors (bigrams, weights, ILP). All of the following experiments use the TAC 2009 data as the test set.

### 4.2 Effect of Bigram Weights

In this experiment, we vary the weighting methods for the two systems: our proposed method and the ICSI system. We use three weighting setups: the estimated bigram frequency value in our method, document frequency, or term frequency from the original text. Table 2 and 3 show the results using the top 100 bigrams from our system and the initial bigram set from the ICSI system respectively. We also evaluate using the two different ILP configurations in these experiments.

First of all, we can see that for both ILP systems, our estimated bigram weights outperform the other frequency-based weights. For the ICSI ILP system, using bigram document frequency achieves better performance than term frequency (which verified why document frequency is used in their system). In contrast, for our ILP method,

#	Weight	ILP	ROUGE-2
1	Estimated value	Proposed	<b>0.1246</b>
2		ICSI	0.1178
3	Document freq	Proposed	0.1109
4		ICSI	0.1132
5	Term freq	Proposed	0.1116
6		ICSI	0.1080

Table 2: Results using different weighting methods on the top 100 bigrams generated from our proposed system.

#	Weight	ILP	ROUGE-2
1	Estimated value	Proposed	0.1157
2		ICSI	<b>0.1161</b>
3	Document freq	Proposed	0.1101
4		ICSI	0.1160
5	Term freq	Proposed	0.1109
6		ICSI	0.1072

Table 3: Results using different weighting methods based on the initial bigram sets. The average number of bigrams is around 80 for each topic.

the bigram’s term frequency is slightly more useful than its document frequency. This indicates that our estimated value is more related to bigram’s term frequency in the original text. When the weight is document frequency, the ICSI’s result is better than our proposed ILP; whereas when using term frequency as the weights, our ILP has better results, again suggesting term frequency fits our ILP system better. When the weight is estimated value, the results depend on the bigram set used. The ICSI’s ILP performs slightly better than ours when it is equipped with the initial bigram, but our proposed ILP has much better results using our selected top100 bigrams. This shows that the size and quality of the bigrams has an impact on the ILP modules.

### 4.3 The Effect of Bigram Set’s size

In our proposed system, we use 100 top bigrams. There are about 80 bigrams used in the ICSI ILP system. A natural question to ask is the impact of the number of bigrams and their quality on the summarization system. Table 4 shows some statistics of the bigrams. We can see that about one third of bigrams in the reference summary are in the original text (127.3 out of 321.93), verifying that people do use different words/bigram when

writing abstractive summaries. We mentioned that we only use the top- $N$  ( $n$  is 100 in previous experiments) bigrams in our summarization system. On one hand, this is to save computational cost for the ILP module. On the other hand, we see from the table that only 127 of these more than 2K bigrams are in the reference summary and are thus expected to help the summary responsiveness. Including all the bigrams would lead to huge noise.

# bigrams in ref summary	321.93
# bigrams in text and ref summary	127.3
# bigrams used in our regression model (i.e., in selected sentences)	2140.7

Table 4: Bigram statistics. The numbers are the average ones for each topic.

Fig 1 shows the bigram coverage (number of bigrams used in the system that are also in reference summaries) when we vary  $N$  selected bigrams. As expected, we can see that as  $n$  increases, there are more reference summary bigrams included in the system. There are 25 summary bigrams in the top-50 bigrams and about 38 in top-100 bigrams. Compared with the ICSI system that has around 80 bigrams in the initial bigram set and 29 in the reference summary, our estimation module has better coverage.

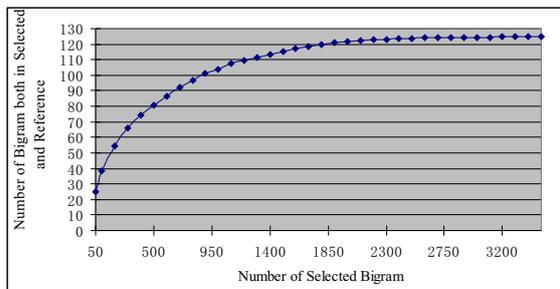


Figure 1: Coverage of bigrams (number of bigrams in reference summary) when varying the number of bigrams used in the ILP systems.

Increasing the number of bigrams used in the system will lead to better coverage, however, the incorrect bigrams also increase and have a negative impact on the system performance. To examine the best tradeoff, we conduct the experiments by choosing the different top- $N$  bigram set for the two ILP systems, as shown in Fig 2. For both the ILP systems, we used the estimated weight value for the bigrams.

We can see that the ICSI ILP system performs better when the input bigrams have less noise (those bigrams that are not in summary). However, our proposed method is slightly more robust to this kind of noise, possibly because of the weights we use in our system – the noisy bigrams have lower weights and thus less impact on the final system performance. Overall the two systems have similar trends: performance increases at the beginning when using more bigrams, and after certain points starts degrading with too many bigrams. The optimal number of bigrams differs for the two systems, with a larger number of bigrams in our method. We also notice that the ICSI ILP system achieved a ROUGE-2 of 0.1218 when using top 60 bigrams, which is better than using the initial bigram set in their method (0.1160).

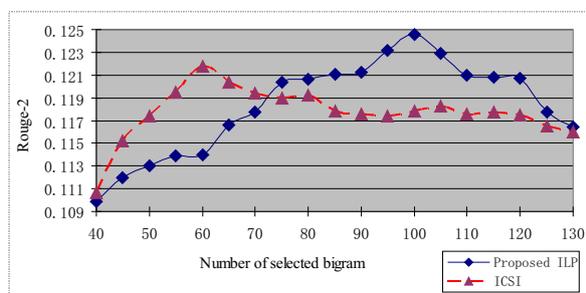


Figure 2: Summarization performance when varying the number of bigrams for two systems.

#### 4.4 Oracle Experiments

Based on the above analysis, we can see the impact of the bigram set and their weights. The following experiments are designed to demonstrate the best system performance we can achieve if we have access to good quality bigrams and weights. Here we use the information from the reference summary.

The first is an oracle experiment, where we use all the bigrams from the reference summaries that are also in the original text. In the ICSI ILP system, the weights are the document frequency from the multiple reference summaries. In our ILP module, we use the term frequency of the bigram. The oracle results are shown in Table 5. We can see these are significantly better than the automatic systems.

From Table 5, we notice that ICSI’s ILP performs marginally better than our proposed ILP. We hypothesize that one reason may be that many bigrams in the summary reference only appear once. Table 6 shows the frequency of the bigrams in the summary. Indeed 85% of bigram only appear once

ILP System	ROUGE-2
Our ILP	0.2124
ICSI ILP	0.2128

Table 5: Oracle experiment: using bigrams and their frequencies in the reference summary as weights.

and no bigrams appear more than 9 times. For the majority of the bigrams, our method and the ICSI ILP are the same. For the others, our system has slight disadvantage when using the reference term frequency. We expect the high term frequency may need to be properly smoothed/normalized.

Freq	1	2	3	4	5	6	7	8	9
Ave#	277	32	7.5	3.2	1.1	0.3	0.1	0.1	0.04

Table 6: Average number of bigrams for each term frequency in one topic’s reference summary.

We also treat the oracle results as the gold standard for extractive summarization and compared how the two automatic summarization systems differ at the sentence level. This is different from the results in Table 1, which are the ROUGE results comparing to human written abstractive summaries at the n-gram level. We found that among the 188 sentences in this gold standard, our system hits 31 and ICSI only has 23. This again shows that our system has better performance, not just at the word level based on ROUGE measures, but also at the sentence level. There are on average 3 different sentences per topic between these two results.

In the second experiment, after we obtain the estimated  $N_{b,ref}$  for every bigram in the selected sentences from our regression model, we only keep those bigrams that are in the reference summary, and use the estimated weights for both ILP modules. Table 7 shows the results. We can consider these as the upper bound the system can achieve if we use the automatically estimated weights for the correct bigrams. In this experiment ICSI ILP’s performance still performs better than ours. This might be attributed to the fact there is less noise (all the bigrams are the correct ones) and thus the ICSI ILP system performs well. We can see that these results are worse than the previous oracle experiments, but are better than using the automatically generated bigrams, again showing the bigram and weight estimation is critical for

summarization.

#	Weight	ILP	ROUGE-2
1	Estimated value	Proposed	0.1888
2		ICSI	0.1942

Table 7: Summarization results when using the estimated weights and only keeping the bigrams that are in the reference summary.

#### 4.5 Effect of Training Set

Since our method uses supervised learning, we conduct the experiment to show the impact of training size. In TAC’s data, each topic has two sets of documents. For set A, the task is a standard summarization, and there are 4 reference summaries, each 100 words long; for set B, it is an update summarization task – the summary includes information not mentioned in the summary from set A. There are also 4 reference summaries, with 400 words in total. Table 8 shows the results on 2009 data when using the data from different years and different sets for training. We notice that when the training data only contains set A, the performance is always better than using set B or the combined set A and B. This is not surprising because of the different task definition. Therefore, for the rest of the study on data size impact, we only use data set A from the TAC data and the DUC data as the training set. In total there are about 233 topics from the two years’ DUC data (06, 07) and three years’ TAC data (08, 10, 11). We incrementally add 20 topics every time (from DUC06 to TAC11) and plot the learning curve, as shown in Fig 3. As expected, more training data results in better performance.

Training Set	# Topics	ROUGE-2
08 Corpus (A)	48	0.1192
08 Corpus (B)	48	0.1178
08 Corpus (A+B)	96	0.1188
10 Corpus (A)	46	0.1174
10 Corpus (B)	46	0.1167
10 Corpus (A+B)	92	0.1170
11 Corpus (A)	44	0.1157
11 Corpus (B)	44	0.1130
11 Corpus (A+B)	88	0.1140

Table 8: Summarization performance when using different training corpora.

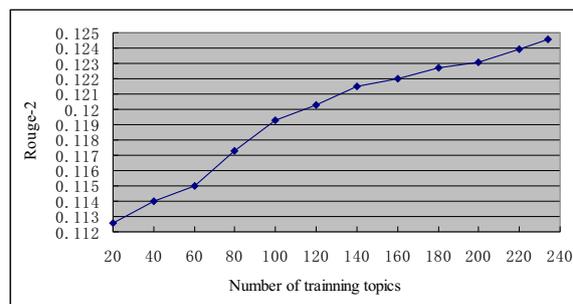


Figure 3: Learning curve

#### 4.6 Summary of Analysis

The previous experiments have shown the impact of the three factors: the quality of the bigrams themselves, the weights used for these bigrams, and the ILP module. We found that the bigrams and their weights are critical for both the ILP setups. However, there is negligible difference between the two ILP methods.

An important part of our system is the supervised method for bigram and weight estimation. We have already seen for the previous ILP method, when using our bigrams together with the weights, better performance can be achieved. Therefore we ask the question whether this is simply because we use supervised learning, or whether our proposed regression model is the key. To answer this, we trained a simple supervised binary classifier for bigram prediction (positive means that a bigram appears in the summary) using the same set of features as used in our bigram weight estimation module, and then used their document frequency in the ICSI ILP system. The result for this method is 0.1128 on the TAC 2009 data. This is much lower than our result. We originally expected that using the supervised method may outperform the unsupervised bigram selection which only uses term frequency information. Further experiments are needed to investigate this. From this we can see that it is not just the supervised methods or using annotated data that yields the overall improved system performance, but rather our proposed regression setup for bigrams is the main reason.

### 5 Related Work

We briefly describe some prior work on summarization in this section. Unsupervised methods have been widely used. In particular, recently several optimization approaches have demonstrated

competitive performance for extractive summarization task. Maximum marginal relevance (MMR) (Carbonell and Goldstein, 1998) uses a greedy algorithm to find summary sentences. (McDonald, 2007) improved the MMR algorithm to dynamic programming. They used a modified objective function in order to consider whether the selected sentence is globally optimal. Sentence-level ILP was also first introduced in (McDonald, 2007), but (Gillick and Favre, 2009) revised it to concept-based ILP. (Woodsend and Lapata, 2012) utilized ILP to jointly optimize different aspects including content selection, surface realization, and rewrite rules in summarization. (Galanis et al., 2012) uses ILP to jointly maximize the importance of the sentences and their diversity in the summary. (Berg-Kirkpatrick et al., 2011) applied a similar idea to conduct the sentence compression and extraction for multiple document summarization. (Jin et al., 2010) made a comparative study on sentence/concept selection and pairwise and list ranking algorithms, and concluded ILP performed better than MMR and the diversity penalty strategy in sentence/concept selection. Other global optimization methods include submodularity (Lin and Bilmes, 2010) and graph-based approaches (Erkan and Radev, 2004; Leskovec et al., 2005; Mihalcea and Tarau, 2004). Various unsupervised probabilistic topic models have also been investigated for summarization and shown promising. For example, (Celikyilmaz and Hakkani-Tür, 2011) used it to model the hidden abstract concepts across documents as well as the correlation between these concepts to generate topically coherent and non-redundant summaries. (Darling and Song, 2011) applied it to separate the semantically important words from the low-content function words.

In contrast to these unsupervised approaches, there are also various efforts on supervised learning for summarization where a model is trained to predict whether a sentence is in the summary or not. Different features and classifiers have been explored for this task, such as Bayesian method (Kupiec et al., 1995), maximum entropy (Osborne, 2002), CRF (Galley, 2006), and recently reinforcement learning (Ryang and Abekawa, 2012). (Aker et al., 2010) used discriminative reranking on multiple candidates generated by A\* search. Recently, research has also been performed to address some issues in the supervised setup, such as the class

data imbalance problem (Xie and Liu, 2010).

In this paper, we propose to incorporate the supervised method into the concept-based ILP framework. Unlike previous work using sentence-based supervised learning, we use a regression model to estimate the bigrams and their weights, and use these to guide sentence selection. Compared to the direct sentence-based classification or regression methods mentioned above, our method has an advantage. When abstractive summaries are given, one needs to use that information to automatically generate reference labels (a sentence is in the summary or not) for extractive summarization. Most researchers have used the similarity between a sentence in the document and the abstractive summary for labeling. This is not a perfect process. In our method, we do not need to generate this extra label for model training since ours is based on bigrams – it is straightforward to obtain the reference frequency for bigrams by simply looking at the reference summary. We expect our approach also paves an easy way for future automatic abstractive summarization. One previous study that is most related to ours is (Conroy et al., 2011), which utilized a Naive Bayes classifier to predict the probability of a bigram, and applied ILP for the final sentence selection. They used more features than ours, whereas we use a discriminatively trained regression model and a modified ILP framework. Our proposed method performs better than their reported results in TAC 2011 data. Another study closely related to ours is (Davis et al., 2012), which leveraged Latent Semantic Analysis (LSA) to produce term weights and selected summary sentences by computing an approximate solution to the Budgeted Maximal Coverage problem.

## 6 Conclusion and Future Work

In this paper, we leverage the ILP method as a core component in our summarization system. Different from the previous ILP summarization approach, we propose a supervised learning method (a discriminatively trained regression model) to determine the importance of the bigrams fed to the ILP module. In addition, we revise the ILP to maximize the bigram gain (which is expected to be highly correlated with ROUGE-2 scores) rather than the concept/bigram coverage. Our proposed method yielded better results than the previous state-of-the-art ILP system on different TAC data

sets. From a series of experiments, we found that there is little difference between the two ILP modules, and that the improved system performance is attributed to the fact that our proposed supervised bigram estimation module can successfully gather the important bigram and assign them appropriate weights. There are several directions that warrant further research. We plan to consider the context of bigrams to better predict whether a bigram is in the reference summary. We will also investigate the relationship between concepts and sentences, which may help move towards abstractive summarization.

## Acknowledgments

This work is partly supported by DARPA under Contract No. HR0011-12-C-0016 and FA8750-13-2-0041, and NSF IIS-0845484. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of DARPA or NSF.

## References

- Ahmet Aker and Robert Gaizauskas. 2009. Summary generation for toponym-referenced images using object type language models. In *Proceedings of the International Conference RANLP*.
- Ahmet Aker, Trevor Cohn, and Robert Gaizauskas. 2010. Multi-document summarization using a\* search and discriminative training. In *Proceedings of the EMNLP*.
- Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the ACL*.
- Ronald Brandow, Karl Mitze, and Lisa F. Rau. 1995. Automatic condensation of electronic publications by sentence selection. *Inf. Process. Manage.*
- Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the SIGIR*.
- Asli Celikyilmaz and Dilek Hakkani-Tür. 2011. Discovery of topically coherent sentences for extractive summarization. In *Proceedings of the ACL*.
- John M. Conroy, Judith D. Schlesinger, Jeff Kubina, Peter A. Rankel, and Dianne P. O’Leary. 2011. Classy 2011 at tac: Guided and multi-lingual summaries and evaluation metrics. In *Proceedings of the TAC*.
- William M. Darling and Fei Song. 2011. Probabilistic document modeling for syntax removal in text summarization. In *Proceedings of the ACL*.
- Sashka T. Davis, John M. Conroy, and Judith D. Schlesinger. 2012. Occams - an optimal combinatorial covering algorithm for multi-document summarization. In *Proceedings of the ICDM*.
- H. P. Edmundson. 1969. New methods in automatic extracting. *J. ACM*.
- Günes Erkan and Dragomir R. Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*
- Dimitrios Galanis, Gerasimos Lampouras, and Ion Androutsopoulos. 2012. Extractive multi-document summarization with integer linear programming and support vector regression. In *Proceedings of the COLING*.
- Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the EMNLP*.
- Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing on NAACL*.
- Dan Gillick, Benoit Favre, and Dilek Hakkani-Tür. 2008. In *The ICSI Summarization System at TAC 2008*.
- Feng Jin, Minlie Huang, and Xiaoyan Zhu. 2010. A comparative study on ranking and selection strategies for multi-document summarization. In *Proceedings of the COLING*.
- Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Proceedings of the SIGIR*.
- Jure Leskovec, Natasa Milic-Frayling, and Marko Grobelnik. 2005. Impact of linguistic analysis on the semantic graph coverage and learning of document extracts. In *Proceedings of the AAAI*.
- Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Proceedings of the NAACL*.
- Chin-Yew Lin. 2004. Rouge: a package for automatic evaluation of summaries. In *Proceedings of the ACL*.
- Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the European conference on IR research*.
- Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the EMNLP*.
- Miles Osborne. 2002. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization*.

- Dragomir R. Radev. 2001. Experiments in single and multidocument summarization using mead. In *In First Document Understanding Conference*.
- Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In *Proceedings of the EMNLP*.
- Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: training log-linear models on unlabeled data. In *Proceedings of the ACL*.
- Kristian Woodsend and Mirella Lapata. 2012. Multiple aspect summarization using integer linear programming. In *Proceedings of the EMNLP*.
- Shasha Xie and Yang Liu. 2010. Improving supervised learning for meeting summarization using sampling and regression. *Comput. Speech Lang.*

# Summarization Through Submodularity and Dispersion

**Anirban Dasgupta**

Yahoo! Labs

Sunnyvale, CA 95054

anirban@yahoo-inc.com

**Ravi Kumar**

Google

Mountain View, CA 94043

tintin@google.com

**Sujith Ravi**

Google

Mountain View, CA 94043

sravi@google.com

## Abstract

We propose a new optimization framework for summarization by generalizing the submodular framework of (Lin and Bilmes, 2011). In our framework the summarization desideratum is expressed as a sum of a submodular function and a non-submodular function, which we call *dispersion*; the latter uses inter-sentence dissimilarities in different ways in order to ensure non-redundancy of the summary. We consider three natural dispersion functions and show that a greedy algorithm can obtain an approximately optimal summary in all three cases. We conduct experiments on two corpora—DUC 2004 and user comments on news articles—and show that the performance of our algorithm outperforms those that rely only on submodularity.

## 1 Introduction

Summarization is a classic text processing problem. Broadly speaking, given one or more documents, the goal is to obtain a concise piece of text that contains the most salient points in the given document(s). Thanks to the omnipresent information overload facing all of us, the importance of summarization is gaining; semi-automatically summarized content is increasingly becoming user-facing: many newspapers equip editors with automated tools to aid them in choosing a subset of user comments to show. Summarization has been studied for the past in various settings—a large single document, multiple documents on the same topic, and user-generated content.

Each domain throws up its own set of idiosyncrasies and challenges for the summarization task. On one hand, in the multi-document case (say, different news reports on the same event), the text is

often very long and detailed. The precision/recall requirements are higher in this domain and a semantic representation of the text might be needed to avoid redundancy. On the other hand, in the case of user-generated content (say, comments on a news article), even though the text is short, one is faced with a different set of problems: volume (popular articles generate more than 10,000 comments), noise (most comments are vacuous, linguistically deficient, and tangential to the article), and redundancy (similar views are expressed by multiple commenters). In both cases, there is a delicate balance between choosing the salient, relevant, popular, and diverse points (e.g., sentences) versus minimizing syntactic and semantic redundancy.

While there have been many approaches to automatic summarization (see Section 2), our work is directly inspired by the recent elegant framework of (Lin and Bilmes, 2011). They employed the powerful theory of submodular functions for summarization: submodularity embodies the “diminishing returns” property and hence is a natural vocabulary to express the summarization desiderata. In this framework, each of the constraints (relevance, redundancy, etc.) is captured as a submodular function and the objective is to maximize their sum. A simple greedy algorithm is guaranteed to produce an approximately optimal summary. They used this framework to obtain the best results on the DUC 2004 corpus.

Even though the submodularity framework is quite general, it has limitations in its expressivity. In particular, it cannot capture redundancy constraints that depend on pairwise dissimilarities between sentences. For example, a natural constraint on the summary is that the sum or the minimum of pairwise dissimilarities between sentences chosen in the summary should be maximized; this, unfortunately, is not a submodular function. We call functions that depend on inter-sentence pair-

wise dissimilarities in the summary as *dispersion* functions. Our focus in this work is on significantly furthering the submodularity-based summarization framework to incorporate such dispersion functions.

We propose a very general graph-based summarization framework that combines a submodular function with a non-submodular dispersion function. We consider three natural dispersion functions on the sentences in a summary: sum of all-pair sentence dissimilarities, the weight of the minimum spanning tree on the sentences, and the minimum of all-pair sentence dissimilarities. These three functions represent three different ways of using the sentence dissimilarities. We then show that a greedy algorithm can obtain approximately optimal summary in each of the three cases; the proof exploits some nice combinatorial properties satisfied by the three dispersion functions. We then conduct experiments on two corpora: the DUC 2004 corpus and a corpus of user comments on news articles. On DUC 2004, we obtain performance that matches (Lin and Bilmes, 2011), without any serious parameter tuning; note that their framework does not have the dispersion function. On the comment corpus, we outperform their method, demonstrating that value of dispersion functions. As part of our methodology, we also use a new structured representation for summaries.

## 2 Related Work

Automatic summarization is a well-studied problem in the literature. Several methods have been proposed for single- and multi-document summarization (Carbonell and Goldstein, 1998; Conroy and O’Leary, 2001; Takamura and Okumura, 2009; Shen and Li, 2010).

Related concepts have also been used in several other scenarios such as query-focused summarization in information retrieval (Daumé and Marcu, 2006), microblog summarization (Sharifi et al., 2010), event summarization (Filatova, 2004), and others (Riedhammer et al., 2010; Qazvinian et al., 2010; Yatani et al., 2011).

Graph-based methods have been used for summarization (Ganesan et al., 2010), but in a different context—using paths in graphs to produce very short abstractive summaries. For a detailed survey on existing automatic summarization techniques and other related topics, see (Kim et al.,

2011; Nenkova and McKeown, 2012).

## 3 Framework

In this section we present the summarization framework. We start by describing a generic objective function that can be widely applied to several summarization scenarios. This objective function is the sum of a monotone submodular *coverage* function and a non-submodular *dispersion* function. We then describe a simple greedy algorithm for optimizing this objective function with provable approximation guarantees for three natural dispersion functions.

### 3.1 Preliminaries

Let  $C$  be a collection of texts. Depending on the summarization application,  $C$  can refer to the set of documents (e.g., newswire) related to a particular topic as in standard summarization; in other scenarios (e.g., user-generated content), it is a collection of comments associated with a news article or a blog post, etc. For each document  $c \in C$ , let  $S(c)$  denote the set of sentences in  $c$ . Let  $U = \cup_{c \in C} S(c)$  be the universe of all sentences; without loss of generality, we assume each sentence is unique to a document. For a sentence  $u \in U$ , let  $C(u)$  be the document corresponding to  $u$ .

Each  $u \in U$  is associated with a *weight*  $w(u)$ , which might indicate, for instance, how similar  $u$  is to the main article (and/or the query, in query-dependent settings). Each pair  $u, v \in U$  is associated with a *similarity*  $s(u, v) \in [0, 1]$ . This similarity can then be used to define an inter-sentence distance  $d(\cdot, \cdot)$  as follows: let  $d'(u, v) = 1 - s(u, v)$  and define  $d(u, v)$  to be the shortest path distance from  $u$  to  $v$  in the graph where the weight of each edge  $(u, v)$  is  $d'(u, v)$ . Note that  $d(\cdot, \cdot)$  is a metric unlike  $d'(\cdot, \cdot)$ , which may not be a metric. (In addition to being intuitive,  $d(\cdot, \cdot)$  being a metric helps us obtain guarantees on the algorithm’s output.) For a set  $S$ , and a point  $u \notin S$ , define  $d(u, S) = \min_{v \in S} d(u, v)$ .

Let  $k > 0$  be fixed. A *summary* of  $U$  is a subset  $S \subseteq U$ ,  $|S| = k$ . Our aim is to find a summary that maximizes

$$f(S) = g(S) + \delta h(S), \quad (1)$$

where  $g(S)$  is the *coverage* function that is non-negative, monotone, and submodular<sup>1</sup>,  $h(S)$  is a

<sup>1</sup>A function  $f : U \rightarrow \Re$  is *submodular* if for every

dispersion function, and  $\delta \geq 0$  is a parameter that can be used to scale the range of  $h(\cdot)$  to be comparable to that of  $g(\cdot)$ .

For two sets  $S$  and  $T$ , let  $P$  be the set of unordered pairs  $\{u, v\}$  where  $u \in S$  and  $v \in T$ . Our focus is on the following dispersion functions: the *sum* measure  $h_s(S, T) = \sum_{\{u, v\} \in P} d(u, v)$ , the *spanning tree* measure  $h_t(S, T)$  given by the cost of the minimum spanning tree of the set  $S \cup T$ , and the *min* measure  $h_m(S, T) = \min_{\{u, v\} \in P} d(u, v)$ . Note that these functions span from considering the entire set of distances in  $S$  to considering only the minimum distance in  $S$ ; also it is easy to construct examples to show that none of these functions is submodular. Define  $h_*(u, S) = h_*(\{u\}, S)$  and  $h_*(S) = h_*(S, S)$ .

Let  $O$  be the optimal solution of the function  $f$ . A summary  $\tilde{S}$  is a  $\gamma$ -approximation if  $f(\tilde{S}) \geq \gamma f(O)$ .

### 3.2 Algorithm

Maximizing (1) is NP-hard even if  $\delta = 0$  or if  $g(\cdot) = 0$  (Chandra and Halldórsson, 2001). For the special case  $\delta = 0$ , since  $g(\cdot)$  is submodular, a classical greedy algorithm obtains a  $(1 - 1/e)$ -approximation (Nemhauser et al., 1978). But if  $\delta > 0$ , since the dispersion function  $h(\cdot)$  is not submodular, the combined objective  $f(\cdot)$  is not submodular as well. Despite this, we show that a simple greedy algorithm achieves a provable approximation factor for (1). This is possible due to some nice structural properties of the dispersion functions we consider.

---

**Algorithm 2** Greedy algorithm, parametrized by the dispersion function  $h$ ; here,  $U, k, g, \delta$  are fixed.

---

```

 $S_0 \leftarrow \emptyset; i \leftarrow 0$ 
for  $i = 0, \dots, k - 1$  do
     $v \leftarrow \arg \max_{u \in U \setminus S_i} g(S_i + u) + \delta h(S_i + u)$ 
     $S_{i+1} \leftarrow S_i \cup \{v\}$ 
end for

```

---

### 3.3 Analysis

In this section we obtain a provable approximation for the greedy algorithm. First, we show that a greedy choice is well-behaved with respect to the dispersion function  $h(\cdot)$ .

**Lemma 1.** *Let  $O$  be any set with  $|O| = k$ . If  $S$  is such that  $|S| = \ell < k$ , then*

$$(i) \sum_{u \in O \setminus S} h_s(u, S) \geq |O \setminus S| \frac{\ell h_s(O)}{k(k-1)};$$

---

$A, B \subseteq U$ , we have  $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$ .

- (ii)  $\sum_{u \in O \setminus S} d(u, S) \geq \frac{1}{2} h_t(O) - h_t(S)$ ; and  
(iii) *there exists  $u \in O \setminus S$  such that  $h_m(u, S) \geq h_m(O)/2$ .*

*Proof.* The proof for (i) follows directly from Lemma 1 in (Borodin et al., 2012).

To prove (ii) let  $T$  be the tree obtained by adding all points of  $O \setminus S$  directly to their respective closest points on the minimum spanning tree of  $S$ .  $T$  is a spanning tree, and hence a Steiner tree, for the points in set  $S \cup O$ . Hence,  $\text{cost}(T) = h_t(S) + \sum_{u \in O \setminus S} d(u, S)$ . Let  $\text{smt}(S)$  denote the cost of a minimum Steiner tree of  $S$ . Thus,  $\text{cost}(T) \geq \text{smt}(O \cup S)$ . Since a Steiner tree of  $O \cup S$  is also a Steiner tree of  $O$ ,  $\text{smt}(O \cup S) \geq \text{smt}(O)$ . Since this is a metric space,  $\text{smt}(O) \geq \frac{1}{2} h_t(O)$  (see, for example, (Cieslik, 2001)). Thus,

$$\begin{aligned} h_t(S) + \sum_{u \in O \setminus S} d(u, S) &\geq \frac{1}{2} h_t(O) \\ \Rightarrow \sum_{u \in O \setminus S} d(u, S) &\geq \frac{1}{2} h_t(O) - h_t(S). \end{aligned}$$

To prove (iii), let  $O = \{u_1, \dots, u_k\}$ . By definition, for every  $i \neq j$ ,  $d(u_i, u_j) \geq h_m(O)$ . Consider the (open) ball  $B_i$  of radius  $h_m(O)/2$  around each element  $u_i$ . By construction for each  $i$ ,  $B_i \cap O = \{u_i\}$  and for each pair  $i \neq j$ ,  $B_i \cap B_j = \emptyset$ . Since  $|S| < k$ , and there are  $k$  balls  $B_i$ , there exists  $k - \ell$  balls  $B_i$  such that  $S \cap B_i = \emptyset$ , proving (iii).  $\square$

We next show that the tree created by the greedy algorithm for  $h = h_t$  is not far from the optimum.

**Lemma 2.** *Let  $u_1, \dots, u_k$  be a sequence of points and let  $S_i = \{u_j, j \leq i\}$ . Then,  $h_t(S_k) \geq \frac{1}{\log k} \sum_{2 \leq j \leq k} d(u_j, S_{j-1})$ .*

*Proof.* The proof follows by noting that we get a spanning tree by connecting each  $u_i$  to its closest point in  $S_{i-1}$ . The cost of this spanning tree is  $\sum_{2 \leq j \leq k} d(u_j, S_{j-1})$  and this tree is also the result of the greedy algorithm run in an online fashion on the input sequence  $\{u_1, \dots, u_k\}$ . Using the result of (Imase and Waxman, 1991), the competitive ratio of this algorithm is  $\log k$ , and hence the proof.  $\square$

We now state and prove the main result about the quality of approximation of the greedy algorithm.

**Theorem 3.** For  $k > 1$ , there is a polynomial-time algorithm that obtains a  $\gamma$ -approximation to  $f(S)$ , where  $\gamma = 1/2$  for  $h = h_s$ ,  $\gamma = 1/4$  for  $h = h_m$ , and  $\gamma = 1/3 \log k$  for  $h = h_t$ .

*Proof.* For  $h_s$  and  $h_t$ , we run Algorithm 1 using a new dispersion function  $h'$ , which is a slightly modified version of  $h$ . In particular, for  $h = h_s$ , we use  $h'(S) = 2h_s(S)$ . For  $h = h_t$ , we abuse notation and define  $h'$  to be a function over an ordered set  $S = \{u_1, \dots, u_k\}$  as follows:  $h'(S) = \sum_{j \leq |S|} d(u_j, S_{j-1})$ , where  $S_{j-1} = \{u_1, \dots, u_{j-1}\}$ . Let  $f'(S) = g(S) + \delta h'(S)$ .

Consider the  $i$ th iteration of the algorithm. By the submodularity of  $g(\cdot)$ ,

$$\begin{aligned} & \sum_{u \in O \setminus S_i} g(S_i \cup \{u\}) - g(S_i) \\ & \geq g(O \cup S_i) - g(S_i) \geq g(O) - g(S_k), \end{aligned} \quad (2)$$

where we use monotonicity of  $g(\cdot)$  to infer  $g(O \cup S_i) \geq g(O)$  and  $g(S_i) \leq g(S_k)$ .

For  $h = h_s$ , the proof follows by Lemma 1(i) and by Theorem 1 in (Borodin et al., 2012).

For  $h_t$ , using the above argument of submodularity and monotonicity of  $g$ , and the result from Lemma 1(ii), we have

$$\begin{aligned} & \sum_{u \in O \setminus S_i} g(S_i \cup u) - g(S_i) + \delta d(u, S_i) \\ & \geq g(O) - g(S_i) + \delta(h_t(O)/2 - h_t(S_i)) \\ & \geq (g(O) + \delta h_t(O)/2) - (g(S_i) + \delta h_t(S_i)) \\ & \geq f(O)/2 - (g(S_i) + \delta h_t(S_i)). \end{aligned}$$

Also,  $h_t(S_i) \leq 2 \text{smt}(S_i)$  since this is a metric space. Using the monotonicity of the Steiner tree cost,  $\text{smt}(S_i) \leq \text{smt}(S_k) \leq h_t(S_k)$ . Hence,  $h_t(S_i) \leq 2h_t(S_k)$ . Thus,

$$\begin{aligned} & \sum_{u \in O \setminus S_i} g(S_i \cup u) - g(S_i) + \delta d(u, S_i) \\ & \geq f(O)/2 - (g(S_i) + \delta h_t(S_i)) \\ & \geq f(O)/2 - (g(S_k) + 2\delta h_t(S_k)) \\ & \geq f(O)/2 - 2f(S_k). \end{aligned} \quad (3)$$

By the greedy choice of  $u_{i+1}$ ,

$$\begin{aligned} & f'(S_i \cup u_{i+1}) - f'(S_i) \\ & = g(S_i \cup u_{i+1}) - g(S_i) + \delta d(u_{i+1}, S_i) \\ & \geq (f(O)/2 - 2f(S_k))/|O \setminus S_i| \\ & \geq \frac{1}{k} (f(O)/2 - 2f(S_k)). \end{aligned}$$

Summing over all  $i \in [1, k-1]$ ,

$$f'(S_k) \geq (k-1)/k (f(O)/2 - 2f(S_k)). \quad (4)$$

Using Lemma 2 we obtain

$$\begin{aligned} f(S_k) & = g(S_k) + \delta h_t(S_k) \geq \frac{f'(S_k)}{\log k} \\ & \geq \frac{1 - 1/k}{\log k} (f(O)/2 - 2f(S_k)). \end{aligned}$$

By simplifying, we obtain  $f(S_k) \geq f(O)/3 \log k$ .

Finally for  $h_m$ , we run Algorithm 1 twice: once with  $g$  as given and  $h \equiv 0$ , and the second time with  $g \equiv 0$  and  $h \equiv h_m$ . Let  $S_g$  and  $S_h$  be the solutions in the two cases. Let  $O_g$  and  $O_h$  be the corresponding optimal solutions. By the submodularity and monotonicity of  $g(\cdot)$ ,  $g(S_g) \geq (1 - 1/e)g(O_g) \geq g(O_g)/2$ . Similarly, using Lemma 1(iii),  $h_m(S_h) \geq h_m(O_h)/2$  since in any iteration  $i < k$  we can choose an element  $u_{i+1}$  such that  $h_m(u_{i+1}, S_i) \geq h_m(O_h)/2$ . Let  $S = \arg \max_{X \in \{S_g, S_h\}} f(X)$ . Using an averaging argument, since  $g$  and  $h_m$  are both non-negative,

$$f(X) \geq (f(S_g) + f(S_h))/2 \geq (g(O_g) + \delta h_m(O_h))/4.$$

Since by definition  $g(O_g) \geq g(O)$  and  $h_m(O_h) \geq h_m(O)$ , we have a  $1/4$ -approximation.  $\square$

### 3.4 A universal constant-factor approximation

Using the above algorithm that we used for  $h_m$ , it is possible to give a universal algorithm that gives a constant-factor approximation to each of the above objectives. By running the Algorithm 1 once for  $g \equiv 0$  and next for  $h \equiv 0$  and taking the best of the two solutions, we can argue that the resulting set gives a constant factor approximation to  $f$ . We do not use this algorithm in our experiments, as it is oblivious of the actual dispersion functions used.

## 4 Using the Framework

Next, we describe how the framework described in Section 3 can be applied to our tasks of interest, i.e., summarizing documents or user-generated content (in our case, comments). First, we represent the elements of interest (i.e., sentences within comments) in a structured manner by using dependency trees. We then use this representation to

generate a graph and instantiate our summarization objective function with specific components that capture the desiderata of a given summarization task.

#### 4.1 Structured representation for sentences

In order to instantiate the summarization graph (nodes and edges), we first need to model each sentence (in multi-document summarization) or comment (i.e., set of sentences) as nodes in the graph. Sentences have been typically modeled using standard ngrams (unigrams or bigrams) in previous summarization work. Instead, we model sentences using a structured representation, i.e., its syntax structure using dependency parse trees. We first use a dependency parser (de Marneffe et al., 2006) to parse each sentence and extract the set of dependency relations associated with the sentence. For example, the sentence “I adore tennis” is represented by the dependency relations (nsubj: adore, I) and (dobj: adore, tennis).

Each sentence represents a single node  $u$  in the graph (unless otherwise specified) and is comprised of a set of dependency relations (or ngrams) present in the sentence. Furthermore, the edge weights  $s(u, v)$  represent pairwise similarity between sentences or comments (e.g., similarity between views expressed in different comments). The edge weights are then used to define the inter-sentence distance metric  $d(u, v)$  for the different dispersion functions. We identify similar views/opinions by computing semantic similarity rather than using standard similarity measures (such as cosine similarity based on exact lexical matches between different nodes in the graph). For each pair of nodes  $(u, v)$  in the graph, we compute the semantic similarity score (using WordNet) between every pair of dependency relation (rel:  $a, b$ ) in  $u$  and  $v$  as:

$$s(u, v) = \sum_{\substack{\text{rel}_i \in u, \text{rel}_j \in v \\ \text{rel}_i = \text{rel}_j}} \text{WN}(a_i, a_j) \times \text{WN}(b_i, b_j),$$

where rel is a relation type (e.g., nsubj) and  $a, b$  are the two arguments present in the dependency relation ( $b$  does not exist for some relations).  $\text{WN}(w_i, w_j)$  is defined as the WordNet similarity score between words  $w_i$  and  $w_j$ .<sup>2</sup> The edge weights are then normalized across all edges in the

<sup>2</sup>There exists various semantic relatedness measures based on WordNet (Patwardhan and Pedersen, 2006). In our experiments, for WN we pick one that is based on the path length between the two words in the WordNet graph.

graph.

This allows us to perform approximate matching of syntactic treelets obtained from the dependency parses using semantic (WordNet) similarity. For example, the sentences “I adore tennis” and “Everyone likes tennis” convey the same view and should be assigned a higher similarity score as opposed to “I hate tennis”. Using the syntactic structure along with semantic similarity helps us identify useful (valid) nuggets of information within comments (or documents), avoid redundancies, and identify similar views in a semantic space.

#### 4.2 Components of the coverage function

Our coverage function is a linear combination of the following.

(i) *Popularity*. One of the requirements for a good summary (especially, for user-generated content) is that it should include (or rather not miss) the popular views or opinions expressed by several users across multiple documents or comments. We model this property in our objective function as follows.

For each node  $u$ , we define  $w(u)$  as the number of documents  $|C_{u_{\text{rel}}} \subseteq C|$  from the collection such that at least one of the dependency relations  $\text{rel} \in u$  appeared in a sentence within some document  $c \in C_{u_{\text{rel}}}$ . The popularity scores are then normalized across all nodes in the graph. We then add this component to our objective function as  $w(S) = \sum_{u \in S} w(u)$ .

(ii) *Cluster contribution*. This term captures the fact that we do not intend to include multiple sentences from the same comment (or document). Define  $\mathcal{B}$  to be the clustering induced by the sentence to comment relation, i.e., two sentences in the same comment belong to the same cluster. The corresponding contribution to the objective function is  $\sum_{B \in \mathcal{B}} |S \cap B|^{1/2}$ .

(iii) *Content contribution*. This term promotes the diversification of content. We look at the graph of sentences where the weight of each edge is  $s(u, v)$ . This graph is then partitioned based on a local random walk based method to give us clusters  $\mathcal{D} = \{D_1, \dots, D_n\}$ . The corresponding contribution to the objective function is  $\sum_{D \in \mathcal{D}} |S \cap D|^{1/2}$ .

(iv) *Cover contribution*. We also measure the cover of the set  $S$  as follows: for each element  $s$  in  $U$  first define cover of an element  $u$  by a set  $S'$  as  $\text{cov}(u, S') = \sum_{v \in S'} s(u, v)$ . Then, the

cover value of the set  $S$  is defined as  $\text{cov}(S) = \sum_{u \in S} \min(\text{cov}(u, S), 0.25\text{cov}(u, U))$ .<sup>3</sup>

Thus, the final coverage function is:  $g(S) = w(S) + \alpha \sum_{B \in \mathcal{B}} |S \cap B|^{1/2} + \beta \sum_{D \in \mathcal{D}} |S \cap D|^{1/2} + \lambda \text{cov}(S)$ , where  $\alpha, \beta, \lambda$  are non-negative constants. By using the monotone submodularity of each of the component functions, and the fact that addition preserves submodularity, the following is immediate.

**Fact 4.**  $g(S)$  is a monotone, non-negative, sub-modular function.

We then apply Algorithm 1 to optimize (1).

## 5 Experiments

### 5.1 Data

**Multi-document summarization.** We use the DUC 2004 corpus<sup>4</sup> that comprises 50 clusters (i.e., 50 different summarization tasks) with 10 documents per cluster on average. Each document contains multiple sentences and the goal is to produce a summary of all the documents for a given cluster.

**Comments summarization.** We extracted a set of news articles and corresponding user comments from Yahoo! News website. Our corpus contains a set of 34 articles and each article is associated with anywhere from 100–500 comments. Each comment contains more than three sentences and 36 words per sentence on average.

### 5.2 Evaluation

For each summarization task, we compare the system output (i.e., summaries automatically produced by the algorithm) against the human-generated summaries and evaluate the performance in terms of ROUGE score (Lin, 2004), a standard recall-based evaluation measure used in summarization. A system that produces higher ROUGE scores generates better quality summary and vice versa.

We use the following evaluation settings in our experiments for each summarization task:

(1) For multi-document summarization, we compute the ROUGE-1<sup>5</sup> scores that was the main evaluation criterion for DUC 2004 evaluations.

<sup>3</sup>The choice of the value 0.25 in the cover component is inspired by the observations made by (Lin and Bilmes, 2011) for the  $\alpha$  value used in their cover function.

<sup>4</sup><http://duc.nist.gov/duc2004/tasks.html>

<sup>5</sup>ROUGE v1.5.5 with options: -a -c 95 -b 665 -m -n 4 -w 1.2

(2) For comment summarization, the collection of user comments associated with a given article is typically much larger. Additionally, individual comments are noisy, wordy, diverse, and informally written. Hence for this task, we use a slightly different evaluation criterion that is inspired from the DUC 2005-2007 summarization evaluation tasks.

We represent the content within each comment  $c$  (i.e., all sentences  $S(c)$  comprising the comment) as a single node in the graph. We then run our summarization algorithm on the instantiated graph to produce a summary for each news article. In addition, each news article and corresponding set of comments were presented to three human annotators. They were asked to select a subset of comments (at most 20 comments) that best represented a summary capturing the most popular as well as diverse set of views and opinions expressed by different users that are relevant to the given news article. We then compare the automatically generated comment summaries against the human-generated summaries and compute the ROUGE-1 and ROUGE-2 scores.<sup>6</sup>

This summarization task is particularly hard for even human annotators since user-generated comments are typically noisy and there are several hundreds of comments per article. Similar to existing work in the literature (Sekine and Nobata, 2003), we computed inter-annotator agreement for the humans by comparing their summaries against each other on a small held-out set of articles. The average ROUGE-1 F-scores observed for humans was much higher (59.7) than that of automatic systems measured against the human-generated summaries (our best system achieved a score of 28.9 ROUGE-1 on the same dataset). This shows that even though this is a new type of summarization task, humans tend to generate more consistent summaries and hence their annotations are reliable for evaluation purposes as in multi-document summarization.

### 5.3 Results

**Multi-document summarization.** (1) Table 1 compares the performance of our system with the previous best reported system that participated in the DUC 2004 competition. We also include for comparison another baseline—a version

<sup>6</sup>ROUGE v1.5.5 with options: -a -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -d -l 150

of our system that approximates the submodular objective function proposed by (Lin and Bilmes, 2011).<sup>7</sup> As shown in the results, our best system<sup>8</sup> which uses the  $h_s$  dispersion function achieves a better ROUGE-1 F-score than all other systems.

(2) We observe that the  $h_m$  and  $h_t$  dispersion functions produce slightly lower scores than  $h_s$ , which may be a characteristic of this particular summarization task. We believe that the empirical results achieved by different dispersion functions depend on the nature of the summarization tasks and there are task settings under which  $h_m$  or  $h_t$  perform better than  $h_s$ . For example, we show later how using the  $h_t$  dispersion function yields the best performance on the comments summarization task. Regardless, the theoretical guarantees presented in this paper cover all these cases.

(3) We also analyze the contributions of individual components of the new objective function towards summarization performance by selectively setting certain parameters to 0. Table 2 illustrates these results. We clearly see that each component (popularity, cluster contribution, dispersion) individually yields a reasonable summarization performance but the best result is achieved by the combined system (row 5 in the table). We also contrast the performance of the full system with and without the dispersion component (row 4 versus row 5). The results show that optimizing for dispersion yields an improvement in summarization performance.

(4) To understand the effect of utilizing syntactic structure and semantic similarity for constructing the summarization graph, we ran the experiments using just the unigrams and bigrams; we obtained a ROUGE-1 F-score of 37.1. Thus, modeling the syntactic structure (using relations extracted

from dependency parse tree) along with computing similarity in semantic spaces (using WordNet) clearly produces an improvement in the summarization quality (+1.4 improvement in ROUGE-1 F-score). However, while the structured representation is beneficial, we observed that dispersion (and other individual components) contribute similar performance gains even when using ngrams alone. So the improvements obtained from the structured representation and dispersion are complementary.

System	ROUGE-1 F
Best system in DUC 2004 (Lin and Bilmes, 2011), no tuning	37.9 37.4 <sup>7</sup>
Our algorithm with $h = h_m$	37.5
$h = h_s$	<b>38.5</b>
$h = h_t$	36.8

Table 1: Performance on DUC 2004.

**Comments summarization.** (1) Table 3 compares the performance of our system against a baseline system that is constructed by picking comments in order of decreasing length, i.e., we first pick the longest comment (comprising the most number of characters), then the next longest comment and so on, to create an ordered set of comments. The intuition behind this baseline is that longer comments contain more content and possibly cover more topics than short ones.

From the table, we observe that the new system (using either dispersion function) outperforms the baseline by a huge margin (+44% relative improvement in ROUGE-1 and much bigger improvements in ROUGE-2 scores). One reason behind the lower ROUGE-2 scores for the baseline might be that while long comments provide more content (in terms of size), they also add noise and irrelevant information to the generated summaries. Our system models sentences using the syntactic structure and semantics and jointly optimizes for multiple summarization criteria (including dispersion) which helps weed out the noise and identify relevant, useful information within the comments thereby producing better quality summaries. The 95% confidence interval scores for the best system on this task is [36.5–46.9].

(2) Unlike the multi-document summarization, here we observe that the  $h_t$  dispersion function yields the best empirical performance for this task. This observation supports our claim that the choice of the specific dispersion function depends

<sup>7</sup>Note that Lin & Bilmes (2011) report a slightly higher ROUGE-1 score (F-score 38.90) on DUC 2004. This is because their system was tuned for the particular summarization task using the DUC 2003 corpus. On the other hand, even without any parameter tuning our method yields good performance, as evidenced by results on the two different summarization tasks. However, since individual components within our objective function are parametrized it is easy to tune them for a specific task or genre.

<sup>8</sup>For the full system, we weight certain parameters pertaining to cluster contributions and dispersion higher ( $\alpha = \beta = \delta = 5$ ) compared to the rest of the objective function ( $\lambda = 1$ ). Lin & Bilmes (2011) also observed a similar finding (albeit via parameter tuning) where weighting the cluster contribution component higher yielded better performance. If the maximum number of sentences/comments chosen were  $k$ , we brought both  $h_s$  and  $h_t$  to the same approximate scale as  $h_m$  by dividing  $h_s$  by  $k(k-1)/2$  and  $h_t$  by  $k-1$ .

Objective function components	ROUGE-1 F
$\alpha = \beta = \lambda = \delta = 0$	35.7
$w(S) = \beta = \lambda = \delta = 0$	35.1
$h = h_s, w(S) = \alpha = \beta = \lambda = 0$	37.1
$\delta = 0$	37.4
$w(S), \alpha, \beta, \lambda, \delta > 0$	<b>38.5</b>

Table 2: Performance with different parameters (DUC).

on the summarization task and that the dispersion functions proposed in this paper have a wider variety of use cases.

(3) Results showing contributions from individual components of the new summarization objective function are listed in Table 4. We observe a similar pattern as with multi-document summarization. The full system using all components outperform all other parameter settings, achieving the best ROUGE-1 and ROUGE-2 scores. The table also shows that incorporating dispersion into the objective function yields an improvement in summarization quality (row 4 versus row 5).

System	ROUGE-1	ROUGE-2
Baseline (decreasing length)	28.9	2.9
Our algorithm with $h = h_m$	39.2	13.2
$h = h_s$	40.9	15.0
$h = h_t$	<b>41.6</b>	<b>16.2</b>

Table 3: Performance on comments summarization.

Objective function components	ROUGE-1	ROUGE-2
$\alpha = \beta = \lambda = \delta = 0$	36.1	9.4
$w(S) = \beta = \lambda = \delta = 0$	32.1	4.9
$h = h_t, w(S) = \alpha = \beta = \lambda = 0$	37.8	11.2
$\delta = 0$	38.0	11.6
$w(S), \alpha, \beta, \lambda, \delta > 0$	<b>41.6</b>	<b>16.2</b>

Table 4: Performance with different parameters (comments).

## 6 Conclusions

We introduced a new general-purpose graph-based summarization framework that combines a submodular coverage function with a non-submodular dispersion function. We presented three natural dispersion functions that represent three different ways of ensuring non-redundancy (using sentence dissimilarities) for summarization and proved that a simple greedy algorithm can obtain an approximately optimal summary in all these cases. Experiments on two different summarization tasks show

that our algorithm outperforms algorithms that rely only on submodularity. Finally, we demonstrated that using a structured representation to model sentences in the graph improves summarization quality.

For future work, it would be interesting to investigate other related developments in this area and perhaps combine them with our approach to see if further improvements are possible. Firstly, it would interesting to see if dispersion offers similar improvements over a tuned version of the submodular framework of Lin and Bilmes (2011). In a very recent work, Lin and Bilmes (2012) demonstrate a further improvement in performance for document summarization by using mixtures of submodular shells. This is an interesting extension of their previous submodular framework and while the new formulation permits more complex functions, the resulting function is still submodular and hence can be combined with the dispersion measures proposed in this paper. A different body of work uses determinantal point processes (DPP) to model subset selection problems and adapt it for document summarization (Kulesza and Taskar, 2011). Note that DPPs use similarity kernels for performing inference whereas our measures are combinatorial and not kernel-representable. While approximation guarantees for DPPs are open, it would be interesting to investigate the empirical gains by combining DPPs with dispersion-like functions.

## Acknowledgments

We thank the anonymous reviewers for their many useful comments.

## References

- Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proc. PODS*, pages 155–166.
- Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. SIGIR*, pages 335–336.
- Barun Chandra and Magnús Halldórsson. 2001. Facility dispersion and remote subgraphs. *J. Algorithms*, 38(2):438–465.
- Dietmar Cieslik. 2001. *The Steiner Ratio*. Springer.

- John M. Conroy and Dianne P. O’Leary. 2001. Text summarization via hidden Markov models. In *Proc. SIGIR*, pages 406–407.
- Hal Daumé, III and Daniel Marcu. 2006. Bayesian query-focused summarization. In *Proc. COLING/ACL*, pages 305–312.
- Marie-Catherine de Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. LREC*, pages 449–454.
- Elena Filatova. 2004. Event-based extractive summarization. In *Proc. ACL Workshop on Summarization*, pages 104–111.
- Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. In *Proc. COLING*.
- Makoto Imase and Bernard M. Waxman. 1991. Dynamic Steiner tree problem. *SIAM J. Discrete Mathematics*, 4(3):369–384.
- Hyun Duk Kim, Kavita Ganesan, Parikshit Sondhi, and ChengXiang Zhai. 2011. Comprehensive review of opinion summarization. Technical report, University of Illinois at Urbana-Champaign.
- Alex Kulesza and Ben Taskar. 2011. Learning determinantal point processes. In *Proc. UAI*, pages 419–427.
- Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proc. ACL*, pages 510–520.
- Hui Lin and Jeff Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Proc. UAI*, pages 479–490.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out: Proc. ACL Workshop*, pages 74–81.
- G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, 14(1):265–294.
- Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 43–76. Springer.
- Siddharth Patwardhan and Ted Pedersen. 2006. Using WordNet-based context vectors to estimate the semantic relatedness of concepts. In *Proc. EACL Workshop on Making Sense of Sense: Bringing Computational Linguistics and Psycholinguistics Together*, pages 1–8.
- Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation summarization through keyphrase extraction. In *Proc. COLING*, pages 895–903.
- Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tür. 2010. Long story short—Global unsupervised models for keyphrase based meeting summarization. *Speech Commun.*, 52(10):801–815.
- Satoshi Sekine and Chikashi Nobata. 2003. A survey for multi-document summarization. In *Proc. HLT-NAACL Workshop on Text Summarization*, pages 65–72.
- Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. 2010. Summarizing microblogs automatically. In *Proc. HLT/NAACL*, pages 685–688.
- Chao Shen and Tao Li. 2010. Multi-document summarization via the minimum dominating set. In *Proc. COLING*, pages 984–992.
- Hiroya Takamura and Manabu Okumura. 2009. Text summarization model based on maximum coverage problem and its variant. In *Proc. EACL*, pages 781–789.
- Koji Yatani, Michael Novati, Andrew Trusty, and Khai N. Truong. 2011. Review spotlight: A user interface for summarizing user-generated reviews using adjective-noun word pairs. In *Proc. CHI*, pages 1541–1550.

# Subtree Extractive Summarization via Submodular Maximization

**Hajime Morita**

Tokyo Institute of Technology, Japan  
morita@lr.pi.titech.ac.jp

**Hiroya Takamura**

Tokyo Institute of Technology, Japan  
takamura@pi.titech.ac.jp

**Ryohei Sasano**

Tokyo Institute of Technology, Japan  
sasano@pi.titech.ac.jp

**Manabu Okumura**

Tokyo Institute of Technology, Japan  
oku@pi.titech.ac.jp

## Abstract

This study proposes a text summarization model that simultaneously performs sentence extraction and compression. We translate the text summarization task into a problem of extracting a set of dependency subtrees in the document cluster. We also encode obligatory case constraints as must-link dependency constraints in order to guarantee the readability of the generated summary. In order to handle the subtree extraction problem, we investigate a new class of submodular maximization problem, and a new algorithm that has the approximation ratio  $\frac{1}{2}(1 - e^{-1})$ . Our experiments with the NTCIR ACLIA test collections show that our approach outperforms a state-of-the-art algorithm.

## 1 Introduction

Text summarization is often addressed as a task of simultaneously performing sentence extraction and sentence compression (Berg-Kirkpatrick et al., 2011; Martins and Smith, 2009). Joint models of sentence extraction and compression have a great benefit in that they have a large degree of freedom as far as controlling redundancy goes. In contrast, conventional two-stage approaches (Zajic et al., 2006), which first generate candidate compressed sentences and then use them to generate a summary, have less computational complexity than joint models. However, two-stage approaches are suboptimal for text summarization. For example, when we compress sentences first, the compressed sentences may fail to contain important pieces of information due to the length limit imposed on each sentence. On the other

hand, when we extract sentences first, an important sentence may fail to be selected, simply because it is long. Enumerating a huge number of compressed sentences is also infeasible. Joint models can prune unimportant or redundant descriptions without resorting to enumeration.

Meanwhile, *submodular maximization* has recently been applied to the text summarization task, and the methods thereof have performed very well (Lin and Bilmes, 2010; Lin and Bilmes, 2011; Morita et al., 2011). Formalizing summarization as a submodular maximization problem has an important benefit in that the problem can be solved by using a greedy algorithm with a performance guarantee.

We therefore decided to formalize the task of simultaneously performing sentence extraction and compression as a submodular maximization problem. That is, we extract subsentences for making the summary directly from all available subsentences in the documents and not in a stepwise fashion. However, there is a difficulty with such a formalization. In the past, the resulting maximization problem has been often accompanied by thousands of linear constraints representing logical relations between words. The existing greedy algorithm for solving submodular maximization problems cannot work in the presence of such numerous constraints although monotone and non-monotone submodular maximization with constraints other than budget constraints have been studied (Lee et al., 2009; Kulik et al., 2009; Gupta et al., 2010). In this study, we avoid this difficulty by reducing the task to one of extracting dependency subtrees from sentences in the source documents. The reduction replaces the difficulty of numerous linear constraints with another difficulty wherein two subtrees can share the same word to-

ken when they are selected from the same sentence, and as a result, the cost of the union of the two subtrees is not always the mere sum of their costs. We can overcome this difficulty by tackling a new class of submodular maximization problem: a budgeted monotone nondecreasing submodular function maximization with a cost function, where the cost of an extraction unit varies depending on what other extraction units are selected. By formalizing the subtree extraction problem as this new maximization problem, we can treat the constraints regarding the grammaticality of the compressed sentences in a straightforward way and use an arbitrary monotone submodular word score function for words including our word score function (shown later). We also propose a new greedy algorithm that solves this new class of maximization problem with a performance guarantee  $\frac{1}{2}(1 - e^{-1})$ .

We evaluated our method on by using it to perform query-oriented summarization (Tang et al., 2009). Experimental results show that it is superior to state-of-the-art methods.

## 2 Related Work

*Submodularity* is formally defined as a property of a set function for a finite universe  $V$ . The function  $f : 2^V \rightarrow \mathbb{R}$  maps a subset  $S \subseteq V$  to a real value. If for any  $S, T \subseteq V$ ,  $f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$ ,  $f$  is called *submodular*. This definition is equivalent to that of *diminishing returns*, which is well known in the field of economics:  $f(S \cup \{u\}) - f(S) \leq f(T \cup \{u\}) - f(T)$ , where  $T \subseteq S \subseteq V$  and  $u$  is an element of  $V$ . Diminishing returns means that the value of an element  $u$  remains the same or decreases as  $S$  becomes larger. This property is suitable for summarization purposes, because the gain of adding a new sentence to a summary that already contains sufficient information should be small. Therefore, many studies have formalized text summarization as a submodular maximization problem (Lin and Bilmes, 2010; Lin and Bilmes, 2011; Morita et al., 2011). Their approaches, however, have been based on sentence extraction. To our knowledge, there is no study that addresses the joint task of simultaneously performing compression and extraction through an approximate submodular maximization with a performance guarantee.

In the field of constrained maximization problems, Kulik et al. (2009) proposed an algorithm that solves the submodular maximization problem

under multiple linear constraints with a performance guarantee  $1 - e^{-1}$  in polynomial time. Although their approach can represent more flexible constraints, we cannot use their algorithm to solve our problem, because their algorithm needs to enumerate many combinations of elements. Integer linear programming (ILP) formulations can represent such flexible constraints, and they are commonly used to model text summarization (McDonald, 2007). Berg-Kirkpatrick et al. (2011) formulated a unified task of sentence extraction and sentence compression as an ILP. However, it is hard to solve large-scale ILP problems exactly in a practical amount of time.

## 3 Budgeted Submodular Maximization with Cost Function

### 3.1 Problem Definition

Let  $V$  be the finite set of all *valid* subtrees in the source documents, where *valid* subtrees are defined to be the ones that can be regarded as grammatical sentences. In this paper, we regard subtrees containing the root node of the sentence as valid. Accordingly,  $V$  denotes a set of all rooted subtrees in all sentences. A subtree contains a set of elements that are units in a dependency structure (e.g., morphemes, words or clauses). Let us consider the following problem of budgeted monotone nondecreasing submodular function maximization with a cost function:  $\max_{S \subseteq V} \{f(S) : c(S) \leq L\}$ , where  $S$  is a summary represented as a set of subtrees,  $c(\cdot)$  is the cost function for the set of subtrees,  $L$  is our budget, and the submodular function  $f(\cdot)$  scores the summary quality. The cost function is not always the sum of the costs of the covered subtrees, but depends on the set of the covered elements by the subtrees. Here, we will assume that the generated summary has to be as long as or shorter than the given summary length limit, as measured by the number of characters. This means the cost of a subtree is the integer number of characters it contains.

$V$  is partitioned into exclusive subsets  $\mathbb{B}$  of valid subtrees, and each subset corresponds to the original sentence from which the valid subtrees derived. However, the cost of a union of subtrees from different sentences is simply the sum of the costs of subtrees, while the cost of a union of subtrees from the same sentence is smaller than the sum of the costs. Therefore, the problem can be represented as follows:

$$\max_{S \subseteq V} \left\{ f(S) : \sum_{B \in \mathbb{B}} c(B \cap S) \leq L \right\}. \quad (1)$$

For example, if we add a subtree  $t$  containing words  $\{w_a, w_b, w_c\}$  to a summary that already covers words  $\{w_a, w_b, w_d\}$  from the same sentence, the additional cost of  $t$  is only  $c(\{w_c\})$  because  $w_a$  and  $w_b$  are already covered<sup>1</sup>.

The problem has two requirements. The first requirement is that the union of valid subtrees is also a valid subtree. The second requirement is that the union of subtrees and a single valid subtree have the same score and the same cost if they cover the same elements. We will refer to the single valid subtree as the *equivalent* subtree of the union of subtrees. These requirements enable us to represent sentence compression as the extraction of subtrees from a sentence. This is because the requirements guarantee that the extracted subtrees represent a sentence.

### 3.2 Greedy Algorithm

We propose Algorithm 1 that solves the maximization problem (Eq.1). The algorithm is based on ones proposed by Khuller et al. (1999) and Krause et al. (2005). Instead of enumerating all candidate subtrees, we use a *local search* to extract the element that has the highest gain per cost. In the algorithm,  $G_i$  indicates a summary set obtained by adding element  $s_i$  to  $G_{i-1}$ .  $U$  means the set of subtrees that are not extracted. The algorithm iteratively adds to the current summary the element  $s_i$  that has the largest ratio of the objective function gain to the additional cost, unless adding it violates the budget constraint. We set a parameter  $r$  that is the scaling factor proposed by Lin and Bilmes (2010). After the loop, the algorithm compares  $G_i$  with the  $\{s^*\}$  that has the largest value of the objective function among all subtrees that are under the budget, and it outputs the summary candidate with the largest value.

Let us analyze the performance guarantee of Algorithm 1<sup>2</sup>.

<sup>1</sup>Each subset  $B$  corresponds to a kind of greedoid constraint.  $V$  implicitly constrains the model such that it can only select valid subtrees from a set of nodes and edges.

<sup>2</sup>Our performance guarantee is lower than that reported by Lin and Bilmes (2010). However, their proof is erroneous. In their proof of Lemma 2, they derive  $\forall u \in S^* \setminus G_{i-1}, \frac{\rho_u(G_{i-1})}{C_u^r} \leq \frac{\rho_{v_i}(G_{i-1})}{C_{v_i}^r}$ , for any  $i(1 \leq i \leq |G|)$ , from line 4 of their Algorithm 1, which selects the densest element out of all available elements. However, the inequality does not hold for  $i$ , for which element  $u$  selected on line 4 is discarded on line 5 of their algorithm. The performance guarantee of their algorithm is actually the same as ours, since

---

**Algorithm 1** Modified greedy algorithm for budgeted submodular function maximization with a cost function .

---

```

1:  $G_0 \leftarrow \phi$ 
2:  $U \leftarrow V$ 
3:  $i \leftarrow 1$ 
4: while  $U \neq \phi$  do
5:    $s_i \leftarrow \arg \max_{s \in U} \frac{f(G_{i-1} \cup \{s\}) - f(G_{i-1})}{c(G_{i-1} \cup \{s\}) - c(G_{i-1})^r}$ 
6:   if  $c(\{s_i\} \cup G_{i-1}) \leq L$  then
7:      $G_i \leftarrow G_{i-1} \cup \{s_i\}$ 
8:      $i \leftarrow i + 1$ 
9:   end if
10:   $U \leftarrow U \setminus \{s_i\}$ 
11: end while
12:  $\bar{s} \leftarrow \arg \max_{s \in V, c(s) \leq L} f(\{s\})$ 
13: return  $G_f = \arg \max_{S \in \{\bar{s}, G_i\}} f(S)$ 

```

---

**Theorem 1** For a normalized monotone submodular function  $f(\cdot)$ , Algorithm 1 has a constant approximation factor when  $r = 1$  as follows:

$$f(G_f) \geq \left( \frac{1}{2}(1 - e^{-1}) \right) f(S^*), \quad (2)$$

where  $S^*$  is the optimal solution and,  $G_f$  is the solution obtained by Greedy Algorithm 1.

**Proof.** See appendix.

### 3.3 Relation with Discrete Optimization

We argue that our optimization problem can be regarded as an extraction of subtrees rooted at a given node from a directed graph, instead of from a tree. Let  $D$  be the set of edges of the directed graph,  $\mathcal{F}$  be a subset of  $D$  that is a subtree. In the field of combinatorial optimization, a pair  $(D, \mathcal{F})$  is a kind of greedoid: *directed branching greedoid* (Schmidt, 1991). A greedoid is a generalization of the matroid concept. However, while matroids are often used to represent constraints on submodular maximization problems (Conforti and Cornuéjols, 1984; Calinescu et al., 2011), greedoids have not been used for that purpose, in spite of their high representation ability. To our knowledge, this is the first study that gives a constant performance guarantee for the submodular maximization under greedoid (non-matroid) constraints.

---

the guarantee  $\frac{1}{2}(1 - e^{-1})$  was already proved by Krause and Guestrin (2005). We show a counterexample. Suppose that  $V$  is  $\{e_1(\text{density } 4:\text{cost } 6), e_2(\text{density } 2:\text{cost } 4), e_3(\text{density } 3:\text{cost } 1), e_4(\text{density } 1:\text{cost } 1)\}$ , and cost limit  $K$  is 10. The optimal solution is  $S^* = \{e_1, e_2\}$ . Their algorithm selects  $e_1, e_3, e_4$  in this order. However the algorithm selects  $e_2$  on line 4 after selecting  $e_3$ , and it drops  $e_2$  on line 5. As a result,  $e_4$  selected by the algorithm does not satisfy the inequality  $\forall u \in S^* \setminus G_{i-1}, \frac{\rho_u(G_{i-1})}{C_u^r} \leq \frac{\rho_{v_i}(G_{i-1})}{C_{v_i}^r}$ .

## 4 Joint Model of Extraction and Compression

We will formalize the unified task of sentence compression and extraction as a budgeted monotone nondecreasing submodular function maximization with a cost function. In this formalization, a valid subtree of a sentence represents a candidate of a compressed sentence. We will refer to all valid subtrees of a given sentence as a *valid set*. A valid set corresponds to all candidates of the compression of a sentence. Note that although we use the valid set in the formalization, we do not have to enumerate all the candidates for each sentence. Since, from the requirements, the union of valid subtrees is also a valid subtree in the valid set, the model can extract one or more subtrees from one sentence, and generate a compressed sentence by merging those subtrees to generate an equivalent subtree. Therefore, the joint model can extract an arbitrarily compressed sentence as a subtree without enumerating all candidates. The joint model can remove the redundant part as well as the irrelevant part of a sentence, because the model simultaneously extracts and compresses sentences. We can approximately solve the subtree extraction problem by using Algorithm 1. On line 5 of the algorithm, the subtree extraction is performed as a local search that finds *maximal density subtrees* from the whole documents. The maximal density subtree is a subtree that has the highest score per cost of subtree. We use a cost function to represent the cost, which indicates the length of word tokens in the subtree.

In this paper, we address the task of summarization of Japanese text by means of sentence compression and extraction. In Japanese, syntactic subtrees that contain the root of the dependency tree of the original sentence often make grammatical sentences. This means that the requirements mentioned in Section 3.1 that a union of valid subtrees is a valid and equivalent tree is often true for Japanese. The root indicates the predicate of a sentence, and it is syntactically modified by other prior words. Some modifying words can be pruned. Therefore, sentence compression can be represented as edge pruning. The linguistic units we extract are *bunsetsu* phrases, which are syntactic chunks often containing a functional word after one or more content words. We will refer to *bunsetsu* phrases as *phrases* for simplicity. Since Japanese syntactic dependency is generally

defined between two phrases, we use the phrases as the nodes of subtrees.

In this joint model, we generate a compressed sentence by extracting an arbitrary subtree from a dependency tree of a sentence. However, not all subtrees are always valid. The sentence generated by a subtree can be unnatural even though the subtree contains the root node of the sentence. To avoid generating such ungrammatical sentences, we need to detect and retain the obligatory dependency relations in the dependency tree. We address this problem by imposing must-link constraints if a phrase corresponds to an obligatory case of the main predicate. We merge obligatory phrases with the predicate beforehand so that the merged nodes make a single large node.

Although we focus on Japanese in this paper, our approach can be applied to English and other languages if certain conditions are satisfied. First, we need a dependency parser of the language in order to represent sentence compression as dependency tree pruning. Moreover, although, in Japanese, obligatory cases distinguish which edges of the dependency tree can be pruned or not, we need another technique to distinguish them in other languages. For example we can distinguish obligatory phrases from optional ones by using semantic role labeling to detect arguments of predicates. The adaptation to other languages is left for future work.

### 4.1 Objective Function

We extract subtrees from sentences in order to solve the query-oriented summarization problem as a unified one consisting of sentence compression and extraction. We thus need to allocate a query relevance score to each node. Off-the-shelf similarity measures such as the cosine similarity of bag-of-words vectors with query terms would allocate scores to the terms that appear in the query, but would give no scores to terms that do not appear in it. With such a similarity, sentence compression extracts nearly only the query terms and fails to contain important information. Instead, we used Query SnowBall (QSB) (Morita et al., 2011) to calculate the query relevance score of each phrase. QSB is a method for query-oriented summarization, which calculates the similarity between query terms and each word by using co-occurrences within the source documents. Although the authors of QSB also provided scores of word pairs to avoid putting excessive penalties

on word overlaps, we do not score word pairs. The score function is *supermodular* as a score function of subtree extraction<sup>3</sup>, because the union of two subtrees can have extra word pairs that are not included in either subtree. If the extra pair has a positive score, the score of the union is greater than the sum of the score of the subtrees. This violates the definition of submodularity, and invalidates the performance guarantee of our algorithms.

We designed our objective function by combining this relevance score with a penalty for redundancy and too-compressed sentences. Important words that describe the main topic should occur multiple times in a good summary. However, excessive overlap undermines the quality of a summary, as do irrelevant words. Therefore, the scores of overlapping words should be lower than those of new words. The behavior can be represented by a submodular objective function that reduces word scores depending on those already included in the summary. Furthermore, a summary consisting of many too-compressed sentences would lack readability. We thus give a positive reward to long sentences. The positive reward leads to a natural summary being generated with fewer sentences and indirectly penalizes too short sentences. Our positive reward for long sentences is represented as

$$reward(S) = c(S) - |S|, \quad (3)$$

where  $c(S)$  is the cost of summary  $S$ , and  $|S|$  is the number of sentences in  $S$ . Since a sentence must contain more than one character, the reward consistently gives a positive score, and gives a higher score to a summary that consists of fewer sentences.

Let  $d$  be the damping rate,  $count_S(w)$  be the number of sentences containing word  $w$  in summary  $S$ ,  $words(S)$  be the set of words included in summary  $S$ ,  $qsb(w)$  be the query relevance score of word  $w$ , and  $\gamma$  be a parameter that adjusts the rate of sentence compression. Our score function for a summary  $S$  is as follows:

$$f(S) = \sum_{w \in words(S)} \left\{ \sum_{i=0}^{count_S(w)-1} qsb(w)d^i \right\} + \gamma reward(S). \quad (4)$$

An optimization problem with this objective function cannot be regarded as an ILP problem because it contains non-linear terms. It is also ad-

<sup>3</sup>The score is still submodular for the purpose of sentence extraction.

vantageous that the submodular maximization can deal with such objective functions. Note that the objective function is such that it can be calculated according to the type of word. Due to the nature of the objective function, we can use dynamic programming to effectively search for the subtree with the maximal density.

## 4.2 Local Search for Maximal Density Subtree

Let us now discuss the local search used on line 5 of Algorithm 1. We will use a fast algorithm to find the maximal density subtree (MDS) of a given sentence for each cost in Algorithm 1.

Consider the objective function Eq. 4. We can ignore the second term of the reward function while looking for the MDS in a sentence because the number of sentences is the same for every MDS in a sentence. That is, the gain function of adding a subtree to a summary can be represented as the sum of gains for words:

$$g(t) = \sum_{w \in t} \{gain_S(w) + freq_t(w)c(w)\gamma\},$$

$$gain_S(w) = qsb(w)d^{count_S(w)},$$

where  $freq_t(w)$  is the number of  $w$ s in subtree  $t$ , and  $gain_S(w)$  is the gain of adding the word  $w$  to the summary  $S$ . Our algorithm is based on *dynamic programming*, and it selects a subtree that maximizes the gain function per cost.

When the word gain is a constant, the algorithm proposed by Hsieh et al. (2010) can be used to find the MDS. We extended this algorithm to work for submodular word gain functions that are not constant. Note that the gain of a word that occurs only once in the sentence, can be treated as a constant. In what follows, we will describe an extended algorithm to find the MDS even if there is word overlap.

For example, let us describe how to obtain the MDS in the case of a binary tree. First let us tackle the case in which the gain is always constant. Let  $n$  be a node in the tree,  $a$  and  $b$  be child nodes of  $n$ ,  $c(n)$  be the cost of  $n$ ,  $mds_a^c$  be the MDS rooted at  $a$  and have cost  $c$ .  $mds_n = \{mds_n^{c(n)}, \dots, mds_n^L\}$  denotes the set of MDSs for each cost and its root node  $n$ . The valid subtrees rooted at  $n$  can be obtained by taking unions of  $n$  with one or both of  $t_1 \in mds_a$  and  $t_2 \in mds_b$ .  $mds_n^c$  is the union that has the largest gain over the union with the cost of  $c$  (by enumerating all the unions). The MDS for

the sentence root can be found by calculating each  $mds_n^c$  from the bottom of the tree to the top.

Next, let us consider the objective function that returns the sum of values of submodular word gain functions. When there is no word overlap within the union, we can obtain  $mds_n^c$  in the same manner as for the constant gain. In contrast, if the union includes word overlap, the gain is less than the sum of gains:  $g(mds_n^c) \leq g(n) + g(mds_a^k) + g(mds_b^{c-k-c(n)})$ , where  $k$  and  $c$  are variables. The score reduction can change the order of the gains of the union. That is, it is possible that another union without word overlaps will have a larger gain. Therefore, the algorithm needs to know whether each  $t \in mds_n$  has the potential to have word overlaps with other MDSs. Let  $\mathcal{O}$  be the set of words that occur twice or more in the sentence on which the local search focuses. The algorithm stores MDS for each  $o \subseteq \mathcal{O}$ , as well as each cost. By storing MDS for each  $o$  and cost as shown in Fig. 1, the algorithm can find MDS with the largest gain over the combinations of subtrees.

Algorithm 2 shows the procedure. In it,  $t$  and  $m$  denote subtrees,  $words(t)$  returns a set of words in the subtree,  $g(t)$  returns the gain of  $t$ ,  $tree(n)$  means a tree consisting of node  $n$ , and  $t \cup m$  denotes the union of subtrees:  $t$  and  $m$ .  $subt$  indicates a set of current maximal density subtrees among the combinations calculated before.  $newt$  indicates a set of temporary maximal density subtrees for the combinations calculated from line 4 to 8.  $subt_{[cost,ws]}$  indicates a element of  $subt$  that has a cost  $cost$  and contains a set of words  $ws$ .  $newt_{[cost,ws]}$  is defined similarly. Line 1 sets  $subt$  to a set consisting of a subtree that indicates node  $n$  itself. The algorithm calculates maximal density subtrees within combinations of the root node  $n$  and MDSs rooted at child nodes of  $n$ . Line 3 iteratively adds MDSs rooted at a next child node to the combinations; the algorithm then calculates MDSs  $newt$  between  $subt$  and the MDSs of the child node. The procedure from line 6 to 8 selects a subtree that has a larger gain from the temporary maximal subtree and the union of  $t$  and  $m$ . The computational complexity of this algorithm is  $O(NC^2)$  when there is no word overlap within the sentence, where  $C$  denotes the cost of the whole sentence, and  $N$  denotes the number of nodes in the sentence. The complexity order is the same as that of the algorithm of Hsieh et al. (2010). When we treat word overlaps, we need to count

---

**Algorithm 2** Algorithm for finding maximal density subtree for each cost: MDSs.

---

**Function:** MDSs

**Require:** root node  $n$

```

1:  $subt_{[c(n),words(n) \cap \mathcal{O}]} = tree(n)$ 
2:  $newt = \phi$ 
3: for  $i \in \text{child node of } n$  do
4:   for  $t \in MDSs(i)$  do
5:     for  $m \in subt$  do
6:        $index = [c(t \cup m), words(t \cup m) \cap \mathcal{O}]$ 
7:        $newt_{index} = \arg \max_{j \in \{newt_{index}, t \cup m\}} g(j)$ 
8:     end for
9:   end for
10:   $subt = newt$ 
11: end for
12: return  $subt$ 

```

---

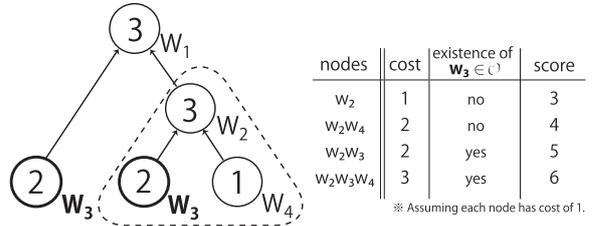


Figure 1: Maximal density subtree extraction. The right table enumerates the subtrees rooted at  $w_2$  in the left tree for all indices. The number in each tree node is the score of the word.

all unions of combinations of the stored MDSs. There are at most  $(C2^{|\mathcal{O}|})$  MDSs that the algorithm needs to store at each node. Therefore the total computational complexity is  $O(NC^22^{2|\mathcal{O}|})$ . Since it is unlikely that a sentence contains many word tokens of one type, the computational cost may not be so large in practical situations.

## 5 Experimental Settings

We evaluate our method on Japanese QA test collections from NTCIR-7 ACLIA1 and NTCIR-8 ACLIA2 (Mitamura et al., 2008; Mitamura et al., 2010). The collections contain questions and weighted answer nuggets. Our experimental settings followed the settings of (Morita et al., 2011), except for the maximum summary length. We generated summaries consisting of 140 Japanese characters or less, with the question as the query terms. We did this because our aim is to use our method in mobile situations. We used “ACLIA1 test data” to tune the parameters, and evaluated our method on “ACLIA2 test” data.

We used JUMAN (Kurohashi and Kawahara, 2009a) for word segmentation and part-of-speech tagging, and we calculated  $idf$  over Mainichi newspaper articles from 1991 to 2005. For the de-

	POURPRE	Precision	Recall	F1	F3
Lin and Bilmes (2011)	0.215	0.126	0.201	0.135	0.174
Subtree extraction (SbE)	0.268	0.238	0.213	<b>0.159</b>	<b>0.190</b>
Sentence extraction (NC)	0.278	0.206	0.215	0.139	0.183

Table 1: Results on ACLIA2 test data.

pendency parsing, we used KNP (Kurohashi and Kawahara, 2009b). Since KNP internally has a flag that indicates either an “obligatory case” or an “adjacent case”, we regarded dependency relations flagged by KNP as obligatory in the sentence compression. KNP utilizes Kyoto University’s case frames (Kawahara and Kurohashi, 2006) as the resource for detecting obligatory or adjacent cases.

To evaluate the summaries, we followed the practices of the TAC summarization tasks (Dang, 2008) and NTCIR ACLIA tasks, and computed pyramid-based precision with the allowance parameter, recall, and  $F_\beta$  (where  $\beta$  is 1 or 3) scores. The allowance parameter was determined from the average nugget length for each question type of the ACLIA2 collection (Mitamura et al., 2010). Precision and recall are computed from the nuggets that the summary covered along with their weights. One of the authors of this paper manually evaluated whether each nugget matched the summary. We also used the automatic evaluation measure, POURPRE (Lin and Demner-Fushman, 2006). POURPRE is based on word matching of reference nuggets and system outputs. We regarded as stopwords the most frequent 100 words in Mainichi articles from 1991 to 2005 (the document frequency was used to measure the frequency). We also set the threshold of nugget matching as 0.5 and binarized the nugget matching, following the previous study (Mitamura et al., 2010). We tuned the parameters by using POURPRE on the development dataset.

Lin and Bilmes (2011) designed a monotone submodular function for query-oriented summarization. Their succinct method performed well in DUC from 2004 to 2007. They proposed a positive diversity reward function in order to define a monotone submodular objective function for generating a non-redundant summary. The diversity reward gives a smaller gain for a biased summary, because it consists of gains based on three clusters and calculates a square root score with respect to each sentence. The reward also contains a score for the similarity of a sentence to the query, for purposes of query-oriented summa-

	Recall	Length	# of nuggets
Subtree extraction	0.213	11,143	100
Reconstructed (RC)	0.228	13,797	108

Table 2: Effect of sentence compression.

riziation. Their objective function also includes a coverage function based on the similarity  $w_{i,j}$  between sentences. In the coverage function min function limits the maximum gain  $\alpha \sum_{i \in V} w_{i,j}$ , which is a small fraction  $\alpha$  of the similarity between a sentence  $j$  and the all source documents. The objective function is the sum of the positive reward  $\mathcal{R}$  and the coverage function  $\mathcal{L}$  over the source documents  $V$ , as follows:

$$\begin{aligned} \mathcal{F}(S) &= \mathcal{L}(S) + \sum_{k=1}^3 \lambda_k \mathcal{R}_{Q,k}(S), \\ \mathcal{L}(S) &= \sum_{i \in V} \min \left\{ \sum_{j \in S} w_{i,j}, \alpha \sum_{k \in V} w_{i,k} \right\}, \\ \mathcal{R}_{Q,k} &= \sum_{c \in C_k} \sqrt{\sum_{j \in S \cup c} \left( \frac{\beta}{N} \sum_{i \in V} w_{i,j} + (1 - \beta) r_{j,Q} \right)}, \end{aligned}$$

where  $\alpha$ ,  $\beta$  and  $\lambda_k$  are parameters, and  $r_{j,Q}$  represents the similarity between sentence  $j$  and query  $Q$ . We tuned the parameters on the development dataset. Lin and Bilmes (2011) used three clusters  $C_k$  with different granularities, which were calculated in advance. We set the granularity to  $(0.2N, 0.15N, 0.05N)$  according to the settings of them, where  $N$  is the number of sentences in a document.

We also regarded as stopwords “教える (tell),” “知る (know),” “何 (what)” and their conjugated forms, which are excessively common in questions. For the query expansion in the baseline, we used Japanese WordNet to obtain synonyms and hypernyms of query terms.

## 6 Results

Table 1 summarizes our results. “Subtree extraction (SbE)” is our method, and “Sentence extraction (NC)” is a version of our method without compression. The NC has the same objective function but only extracts sentences. The F1-measure and F3-measure of our method are 0.159 and 0.190 respectively, while those of the state-of-

the-art baseline are 0.135 and 0.174 respectively. Unfortunately, since the document set is small, the difference is not statistically significant. Comparing our method with the one without compression, we can see that there are improvements in the F1 and F3 scores of the human evaluation, whereas the POURPRE score of the version of our method without compression is higher than that of our method with compression. The compression improved the precision of our method, but slightly decreased the recall.

For the error analyses, we reconstructed the original sentences from which our method extracted the subtrees. Table 2 shows the statistics of the summaries of SbE and reconstructed summaries (RC). The original sentences covered 108 answer nuggets in total, and 8 of these answer nuggets were dropped by the sentence compression. Comparing the results of SbE and RC, we can see that the sentence compression caused the recall of SbE to be 7% lower than that of RC. However, the drop is relatively small in light of the fact that the sentence compression can discard 19% of the original character length with SbE. This suggests that the compression can efficiently prune words while avoiding pruning informative content.

Since the summary length is short, we can select only two or three sentences for a summary. As Morita et al. (2011) mentioned, answer nuggets overlap each other. The baseline objective function  $\mathcal{R}$  tends to extract sentences from various clusters. If the answer nuggets are present in the same cluster, the objective function does not fit the situation. However, our methods (SbE and NC) have a parameter  $d$  that can directly adjust overlap penalty with respect to word importance as well as query relevance. This may help our methods to cover similar answer nuggets. In fact, the development data resulted in a relatively high parameter  $d$  (0.8) for NC compared with 0.2 for SbE.

## 7 Conclusions and Future Work

We formalized a query-oriented summarization, which is a task in which one simultaneously performs sentence compression and extraction, as a new optimization problem: budgeted monotone nondecreasing submodular function maximization with a cost function. We devised an approximate algorithm to solve the problem in a reasonable computational time and proved that its approximation rate is  $\frac{1}{2}(1 - e^{-1})$ . Our approach achieved

an F3-measure of 0.19 on the ACLIA2 Japanese test collection, which is 9.2 % improvement over a state-of-the-art method using a submodular objective function.

Since our algorithm requires that the objective function is the sum of word score functions, our proposed method has a restriction that we cannot use an arbitrary monotone submodular function as the objective function for the summary. Our future work will improve the local search algorithm to remove this restriction. As mentioned before, we also plan to adapt of our system to other languages.

## Appendix

Here, we analyze the performance guarantee of Algorithm 1. We use the following notation.  $S^*$  is the optimal solution,  $c_u(S)$  is the residual cost of subtree  $u$  when  $S$  is already covered, and  $i^*$  is the last step before the algorithm discards a subtree  $s \in S^*$  or a part of the subtree  $s$ . This is because the subtree does not belong to either the approximate solution or the optimal solution. We can remove the subtree  $s'$  from  $V$  without changing the approximate rate.  $s_i$  is the  $i$ -th subtree obtained by line 5 of Algorithm 1.  $G_i$  is the set obtained after adding subtree  $s_i$  to  $G_{i-1}$  from the valid set  $B_i$ .  $G_f$  is the final solution obtained by Algorithm 1.  $f(\cdot) : 2^V \rightarrow \mathbb{R}$  is a monotone submodular function.

We assume that there is an equivalent subtree with any union of subtrees in a valid set  $B$ :  $\forall t_1, t_2, \exists t_e, t_e \equiv \{t_1, t_2\}$ . Note that for any order of the set, the cost or profit of the set is fixed:  $\sum_{u_i \in S = \{u_1, \dots, u_{|S|}\}} c_{u_i}(S_{i-1}) = c(S)$ .

**Lemma 1**  $\forall X, Y \subseteq V, f(X) \leq f(Y) + \sum_{u \in X \setminus Y} \rho_u(Y)$ , where  $\rho_u(S) = f(S \cup \{u\}) - f(S)$ .

The inequality can be derived from the definition of submodularity.  $\square$

**Lemma 2** For  $i = 1, \dots, i^* + 1$ , when  $0 \leq r \leq 1$ ,

$$f(S^*) - f(G_{i-1}) \leq \frac{L^r |S^*|^{1-r}}{c_{s_i}(G_{i-1})} (f(G_{i-1} \cup \{s_i\}) - f(G_{i-1})),$$

where  $c_u(S) = c(S \cup \{u\}) - c(S)$ .

*Proof.* From line 5 of Algorithm 1, we have

$$\forall u \in S^* \setminus G_{i-1}, \frac{\rho_u(G_{i-1})}{c_u(G_{i-1})^r} \leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r}.$$

Let  $B$  be a valid set, and  $union$  be a function that returns the union of subtrees. We have

$\forall T \subseteq B, \exists b \in B, b = \text{union}(T)$ , because we have an equivalent tree  $b \in B$  for each union of trees  $T$  in a valid set  $B$ . That is, for any set of subtrees, we have an equivalent set of subtrees, where  $b_i \in B_i$ . Without loss of generality, we can replace the difference set  $S^* \setminus G_{i-1}$  with a set  $T'_{i-1} = \{b_0, \dots, b_{|T'_{i-1}|}\}$  that does not contain any two elements extracted from the same valid set. Thus when  $0 \leq r \leq 1$  and  $0 \leq i \leq i^* + 1$ ,  $\frac{\rho_{S^* \setminus G_{i-1}}(G_{i-1})}{c_{S^* \setminus G_{i-1}}(G_{i-1})^r} = \frac{\rho_{T'_{i-1}}(G_{i-1})}{c_{T'_{i-1}}(G_{i-1})^r}$ , and  $\forall b_j \in T'_{i-1}, \frac{\rho_{b_j}(G_{i-1})}{c_{b_j}(G_{i-1})^r} \leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r}$ . Thus,

$$\begin{aligned} \rho_{T'_{i-1}}(G_{i-1}) &= \sum_{u \in T'_{i-1}} \rho_u(G_{i-1}) \\ &\leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r} \sum_{u \in T'_{i-1}} c_u(G_{i-1})^r \\ &\leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r} |T'_{i-1}| \left( \frac{\sum_{u \in T'_{i-1}} c_u(G_{i-1})}{|T'_{i-1}|} \right)^r \\ &\leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r} |T'_{i-1}|^{1-r} \left( \sum_{u \in T'_{i-1}} c_u(\phi) \right)^r \\ &\leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r} |S^*|^{1-r} L^r, \end{aligned}$$

where the second inequality is from Hölder's inequality. The third inequality uses the submodularity of the cost function,

$$c_u(G_{i-1}) = c(\{u\} \cup G_{i-1}) - c(G_{i-1}) \leq c_u(\phi)$$

and the fact that  $|S^*| \geq |S^* \setminus G_{i-1}| \geq |T'_{i-1}|$ , and  $\sum_{u \in T'_{i-1}} c_u(\phi) = c(T'_{i-1}) \leq L$ .

As a result, we have

$$\begin{aligned} \rho_{S^* \setminus G_{i-1}}(G_{i-1}) &= \rho_{T'_{i-1}}(G_{i-1}) \\ &\leq \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r} |S^*|^{1-r} L^r. \end{aligned}$$

Let  $X = S^*$  and  $Y = G_{i-1}$ . Applying Lemma 1 yields

$$\begin{aligned} f(S^*) &\leq f(G_{i-1}) + \rho_{u \in S^* \setminus G_{i-1}}(G_{i-1}). \\ &\leq f(G_{i-1}) + \frac{\rho_{s_i}(G_{i-1})}{c_{s_i}(G_{i-1})^r} |S^*|^{1-r} L^r. \end{aligned}$$

The lemma follows as a result.

**Lemma 3** For a normalized monotone submodular  $f(\cdot)$ , for  $i = 1, \dots, i^* + 1$  and  $0 \leq r \leq 1$  and letting  $s_i$  be the  $i$ -th unit added into  $G$  and  $G_i$  be the set after adding  $s_i$ , we have

$$f(G_i) \geq \left( 1 - \prod_{k=1}^i \left( 1 - \frac{c_{s_k}(G_{k-1})^r}{L^r |S^*|^{1-r}} \right) \right) f(S^*).$$

*Proof.* This is proved similarly to Lemma 3 of (Krause and Guestrin, 2005) using Lemma 2.

*Proof of Theorem 1.* This is proved similarly to Theorem 1 of (Krause and Guestrin, 2005) using Lemma 3.

## References

- Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 481–490, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Calinescu Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. 2011. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766.
- Michele Conforti and Gérard Cornuéjols. 1984. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete Applied Mathematics*, 7(3):251 – 274.
- Hoang Trang Dang. 2008. Overview of the tac 2008 opinion question answering and summarization tasks. In *Proceedings of Text Analysis Conference*.
- Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. 2010. Constrained non-monotone submodular maximization: offline and secretary algorithms. In *Proceedings of the 6th international conference on Internet and network economics*, WINE'10, pages 246–257, Berlin, Heidelberg. Springer-Verlag.
- Sun-Yuan Hsieh and Ting-Yu Chou. 2010. The weight-constrained maximum-density subtree problem and related problems in trees. *The Journal of Supercomputing*, 54(3):366–380, December.
- Daisuke Kawahara and Sadao Kurohashi. 2006. A fully-lexicalized probabilistic model for japanese syntactic and case structure analysis. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 176–183, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Samir Khuller, Anna Moss, and Joseph S. Naor. 1999. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45.
- Andreas Krause and Carlos Guestrin. 2005. A note on the budgeted maximization on submodular functions. Technical Report CMU-CALD-05-103, Carnegie Mellon University.

- Ariel Kulik, Hadas Shachnai, and Tami Tamir. 2009. Maximizing submodular set functions subject to multiple linear constraints. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 545–554, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.
- Sadao Kurohashi and Daisuke Kawahara, 2009a. *Japanese Morphological Analysis System JUMAN 6.0 Users Manual*. <http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN>.
- Sadao Kurohashi and Daisuke Kawahara, 2009b. *KN parser (Kurohashi-Nagao parser) 3.0 Users Manual*. <http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP>.
- Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. 2009. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC '09, pages 323–332, New York, NY, USA. ACM.
- Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 912–920, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 510–520, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jimmy Lin and Dina Demner-Fushman. 2006. Methods for automatically evaluating answers to complex questions. *Information Retrieval*, 9(5):565–587, November.
- André F. T. Martins and Noah A. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, ILP '09, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European conference on IR research*, ECIR'07, pages 557–564, Berlin, Heidelberg. Springer-Verlag.
- Teruko Mitamura, Eric Nyberg, Hideki Shima, Tsuneaki Kato, Tatsunori Mori, Chin-Yew Lin, Ruihua Song, Chuan-Jie Lin, Tetsuya Sakai, Donghong Ji, and Noriko Kando. 2008. Overview of the NTCIR-7 ACLIA Tasks: Advanced Cross-Lingual Information Access. In *Proceedings of the 7th NTCIR Workshop*.
- Teruko Mitamura, Hideki Shima, Tetsuya Sakai, Noriko Kando, Tatsunori Mori, Koichi Takeda, Chin-Yew Lin, Ruihua Song, Chuan-Jie Lin, and Cheng-Wei Lee. 2010. Overview of the ntcir-8 aclia tasks: Advanced cross-lingual information access. In *Proceedings of the 8th NTCIR Workshop*.
- Hajime Morita, Tetsuya Sakai, and Manabu Okumura. 2011. Query snowball: a co-occurrence-based approach to multi-document summarization for question answering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 223–229, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Wolfgang Schmidt. 1991. Greedoids and searches in directed graphs. *Discrete Mathematics*, 93(1):75–88, November.
- Jie Tang, Limin Yao, and Dewei Chen. 2009. Multi-topic based query-oriented summarization. In *Proceedings of 2009 SIAM International Conference Data Mining (SDM'2009)*, pages 1147–1158.
- David M. Zajic, Bonnie J. Dorr, Jimmy Lin, and Richard Schwartz. 2006. Sentence compression as a component of a multi-document summarization system. In *Proceedings of the 2006 Document Understanding Conference (DUC 2006) at NLT/NAACL 2006*.

# The effect of non-tightness on Bayesian estimation of PCFGs

**Shay B. Cohen**

Department of Computer Science  
Columbia University  
scohen@cs.columbia.edu

**Mark Johnson**

Department of Computing  
Macquarie University  
mark.johnson@mq.edu.au

## Abstract

Probabilistic context-free grammars have the unusual property of not always defining tight distributions (i.e., the sum of the “probabilities” of the trees the grammar generates can be less than one). This paper reviews how this non-tightness can arise and discusses its impact on Bayesian estimation of PCFGs. We begin by presenting the notion of “almost everywhere tight grammars” and show that linear CFGs follow it. We then propose three different ways of reinterpreting non-tight PCFGs to make them tight, show that the Bayesian estimators in Johnson et al. (2007) are correct under one of them, and provide MCMC samplers for the other two. We conclude with a discussion of the impact of tightness empirically.

## 1 Introduction

Probabilistic Context-Free Grammars (PCFGs) play a special role in computational linguistics because they are perhaps the simplest probabilistic models of hierarchical structures. Their simplicity enables us to mathematically analyze their properties to a detail that would be difficult with linguistically more accurate models. Such analysis is useful because it is reasonable to expect more complex models to exhibit similar properties as well.

The problem of inferring PCFG rule probabilities from training data consisting of yields or strings alone is interesting from both cognitive and engineering perspectives. Cognitively it is implausible that children can perceive the parse trees of the language they are learning, but it is more reasonable to assume that they can obtain the terminal strings or yield of these trees. Unsupervised methods for learning a grammar from terminal strings alone is also interesting from an engineering perspective because such training data is cheap and

plentiful, while the manually parsed data required by supervised methods are expensive to produce and relatively rare.

Cohen and Smith (2012) show that inferring PCFG rule probabilities from strings alone is computationally intractable, so we should not expect to find an efficient, general-purpose algorithm for the unsupervised problem. Instead, approximation algorithms are standardly used. For example, the Inside-Outside (IO) algorithm efficiently implements the Expectation-Maximization (EM) procedure for approximating a Maximum Likelihood estimator (Lari and Young, 1990). Bayesian estimators for PCFG rule probabilities have also been attracting attention because they provide a theoretically-principled way of incorporating prior information. Kurihara and Sato (2006) proposed a Variational Bayes estimator based on a mean-field approximation, and Johnson et al. (2007) proposed MCMC samplers for the posterior distribution over rule probabilities and the parse trees of the training data strings.

PCFGs have the interesting property (which we expect most linguistically more realistic models to also possess) that the distributions they define are not always properly normalized or “tight”. In a non-tight PCFG the partition function (i.e., sum of the “probabilities” of all the trees generated by the PCFG) is less than one. (Booth and Thompson, 1973, called such non-tight PCFGs “inconsistent”, but we follow Chi and Geman (1998) in calling them “non-tight” to avoid confusion with the consistency of statistical estimators). Chi (1999) showed that renormalized non-tight PCFGs (which he called “Gibbs CFGs”) define the same class of distributions over trees as do tight PCFGs with the same rules, and provided an algorithm for mapping any PCFG to a tight PCFG with the same rules that defines the same distribution over trees.

An obvious question is then: how does tightness affect the inference of PCFGs? Chi and Geman (1998) studied the question for Maximum Likelihood (ML) estimation, and showed that ML es-

timates are always tight for both the supervised case (where the input consists of parse trees) and the unsupervised case (where the input consists of yields or terminal strings). This means that ML estimators can simply ignore issues of tightness, and rest assured that the PCFGs they estimate are in fact tight.

The situation is more subtle with Bayesian estimators. We show that for the special case of linear PCFGs (which include HMMs) with non-degenerate priors the posterior puts zero mass on non-tight PCFGs, so tightness is not an issue with Bayesian estimation of such grammars. However, because all of the commonly used priors (such as the Dirichlet or the logistic normal) assign non-zero probability across the whole probability simplex, in general the posterior may assign non-zero probability to non-tight PCFGs. We discuss three different possible approaches to this in this paper:

1. the *only-tight* approach, where we modify the prior so it only assigns non-zero probability to tight PCFGs,
2. the *renormalization* approach, where we renormalize non-tight PCFGs so they define a probability distribution over trees, and
3. the *sink-element* approach, where we reinterpret non-tight PCFGs as assigning non-zero probability to a “sink element”, so both tight and non-tight PCFGs are properly normalized.

We show how to modify the Gibbs sampler described by Johnson et al. (2007) so it produces samples from the posterior distributions defined by the only-tight and renormalization approaches. Perhaps surprisingly, we show that Gibbs sampler as defined by Johnson et al. actually produces samples from the posterior distributions defined by the sink-element approach.

We conclude by studying the effect of requiring tightness on the estimation of some simple PCFGs. Because the Bayesian posterior converges around the (tight) ML estimate as the size of the data grows, requiring tightness only seems to make a difference with highly biased priors or with very small training corpora.

## 2 PCFGs and tightness

Let  $G = (T, N, S, R)$  be a Context-Free Grammar in Chomsky normal form with no useless productions, where  $T$  is a finite set of *terminal symbols*,

$N$  is a finite set of *nonterminal symbols* (disjoint from  $T$ ),  $S \in N$  is a distinguished nonterminal called the *start symbol*, and  $R$  is a finite set of *productions* of the form  $A \rightarrow BC$  or  $A \rightarrow w$ , where  $A, B, C \in N$  and  $w \in T$ . In what follows we use  $\beta$  as a variable ranging over  $(N \times N) \cup T$ .

A *Probabilistic Context-Free Grammar*  $(G, \Theta)$  is a pair consisting of a context-free grammar  $G$  and a real-valued vector  $\Theta$  of length  $|R|$  indexed by productions, where  $\theta_{A \rightarrow \beta}$  is the *production probability* associated with the production  $A \rightarrow \beta \in R$ . We require that  $\theta_{A \rightarrow \beta} \geq 0$  and that for all nonterminals  $A \in N$ ,  $\sum_{A \rightarrow \beta \in R_A} \theta_{A \rightarrow \beta} = 1$ , where  $R_A$  is the subset of rules  $R$  expanding the nonterminal  $A$ .

A PCFG  $(G, \Theta)$  defines a measure  $\mu_\Theta$  over trees  $t$  as follows:

$$\mu_\Theta(t) = \prod_{r \in R} \theta_r^{f_r(t)}$$

where  $f_r(t)$  is the number of times the production  $r = A \rightarrow \beta \in R$  is used in the derivation of  $t$ .

The *partition function*  $Z$  or measure of all possible trees is:

$$Z(\Theta) = \sum_{t' \in \mathcal{T}} \prod_{r \in R} \theta_r^{f_r(t')}$$

where  $\mathcal{T}$  is the set of all (finite) trees generated by  $G$ . A PCFG is *tight* iff the partition function  $Z(\Theta) = 1$ . In this paper we use  $\Theta^\perp$  to denote the set of rule probability vectors  $\Theta$  for which  $G$  is non-tight. Nederhof and Satta (2008) survey several algorithms for computing  $Z(\Theta)$ , and hence for determining whether a PCFG is tight.<sup>1</sup>

Non-tightness can arise in very simple PCFGs, such as the “Catalan” PCFG  $S \rightarrow SS \mid a$ . This grammar produces binary trees where all internal nodes are labeled as  $S$  and the yield of these trees is a sequence of  $as$ . If the probability of the rule  $S \rightarrow SS$  is greater than 0.5 then this PCFG is non-tight.

Perhaps the most straight-forward way to understand this non-tightness is to view this grammar as defining a branching process where an  $S$  can either “reproduce” with probability  $\theta_{S \rightarrow SS}$  or “die out”

<sup>1</sup>We found out that finding whether a PCFG is tight by directly inspecting the partition function value is less stable than using the method in Wetherell (1980). For this reason, we used Wetherell’s approach, which is based on finding the principal eigenvalue of the matrix  $M$ .

with probability  $\theta_{S \rightarrow a}$ . When  $\theta_{S \rightarrow S} > \theta_{S \rightarrow a}$  the  $S$  nodes reproduce at a faster rate than they die out, so the derivation has a non-zero probability of endlessly rewriting (Atherya and Ney, 1972).

### 3 Bayesian inference for PCFGs

The goal of Bayesian inference for PCFGs is to infer a posterior distribution over the rule probability vectors  $\Theta$  given observed data  $D$ . This posterior distribution is obtained by combining the likelihood  $P(D | \Theta)$  with a prior distribution  $P(\Theta)$  over  $\Theta$  using Bayes Rule.

$$P(\Theta | D) \propto P(D | \Theta) P(\Theta)$$

We now formally define the three approaches to handling non-tightness mentioned earlier:

**the only-tight approach:** we only permit priors where  $P(\Theta^\perp) = 0$ , i.e., we insist that the prior assign zero mass to non-tight rule probability vectors, so  $Z = 1$ . This means we can define:

$$P(t | \Theta) = \mu_\Theta(t)$$

**the renormalization approach:** we renormalize non-tight PCFGs by dividing by the partition function:

$$P(t | \Theta) = \frac{1}{Z(\Theta)} \mu_\Theta(t) \quad (1)$$

**the sink-element approach:** we redefine our probability distribution so its domain is a set  $\mathcal{T}' = \mathcal{T} \cup \{\perp\}$ , where  $\mathcal{T}$  is the set of (finite) trees generated by  $G$  and  $\perp \notin \mathcal{T}$  is a new element that serves as a “sink state” to which the “missing mass”  $1 - Z(\Theta)$  is assigned. Then we define:<sup>2</sup>

$$P(t | \Theta) = \begin{cases} \mu_\Theta(t) & \text{if } t \in \mathcal{T} \\ 1 - Z(\Theta) & \text{if } t = \perp \end{cases}$$

<sup>2</sup>This definition of a distribution over trees can be induced by a tight PCFG with a special  $\perp$  symbol in its vocabulary. Given  $G$ , the first step is to create a tight grammar  $G_0$  using the renormalization approach. Then, a new start symbol is added to  $G_0$ ,  $S_0$ , and also rules  $S_0 \rightarrow S$  (where  $S$  is the old start symbol in  $G_0$ ) and  $S_0 \rightarrow \perp$ . The first rule is given probability  $Z(\Theta)$  and the second rule is given probability  $1 - Z(\Theta)$ . It can be then readily shown that the new tight PCFG  $G_0$  induces a distribution over trees just like in Eq. 3, only with additional  $S_0$  on top of all trees.

With this in hand, we can now define the likelihood term. We consider two types of data  $D$  here. In the *supervised setting* the data  $D$  consists of a corpus of parse trees  $D = (t_1, \dots, t_n)$  where each tree  $t_i$  is generated by the PCFG  $G$ , so

$$P(D | \Theta) = \prod_{i=1}^n P(t_i | \Theta)$$

In the *unsupervised setting* the data  $D$  consists of a corpus of strings  $D = (w_1, \dots, w_n)$  where each string  $w_i$  is the yield of one or more trees generated by  $G$ . In this setting

$$P(D | \Theta) = \prod_{i=1}^n P(w_i | \Theta), \text{ where:}$$

$$P(w | \Theta) = \sum_{t \in \mathcal{T}: \text{yield}(t)=w} P(t | \Theta)$$

### 4 The special case of linear PCFGs

One way to handle the issue of tightness is to identify a family of CFGs for which practically any parameter setting will yield a tight PCFG. This is the focus of this section, in which we identify a subset of CFGs, which are “almost everywhere” tight. This family of CFGs includes many of the CFGs used in NLP applications.

We cannot expect that a CFG will yield a tight PCFG for *any* assignment to the rule probabilities (i.e. that  $\Theta^\perp = \emptyset$ ). Even in simple cases, such as the grammar  $S \rightarrow S|a$ , the assignment of probability 1 to  $S \rightarrow S$  and 0 to the other rule renders the  $S$  nonterminal useless, and places all of the probability mass on infinite structures of the form  $S \rightarrow S \rightarrow S \rightarrow \dots$

However, we can weaken our requirement so that the cases in which parameter assignment yields a non-tight PCFG are rare, or have measure zero. To put it more formally, we say that a prior  $P(\Theta)$  is “tight almost everywhere for  $G$ ” if

$$P(\Theta^\perp) = \int_{\Theta \in \Theta^\perp} P(\Theta) d\Theta = 0.$$

We now provide a sufficient condition (linearity) for CFGs under which they are tight almost everywhere with any continuous prior.

For a nonterminal  $A \in N$  and  $\beta \in (N \cup T)^*$ , we use  $A \Rightarrow^k \beta$  to denote that  $A$  can be re-written using a sequence of rules from  $R$  to the sentential form  $\beta$  in  $k$  derivation steps. We use  $A \Rightarrow^+ \beta$  to denote that there exists a  $k > 0$  such that  $A \Rightarrow^k \beta$ .

**Definition 1** A context-free grammar  $G$  is linear if there are no  $A \in N$  such that

$$A \Rightarrow^+ \dots A \dots A \dots$$

**Definition 2** A nonterminal  $A \in N$  in a probabilistic context-free grammar  $G$  with parameters  $\Theta$  is nonterminating if

$$P_G(A \Rightarrow^+ \dots A \dots | \Theta) = 1.$$

Here  $P(A \Rightarrow^+ \dots A \dots | \Theta)$  is defined as:

$$\sum_{\beta: \beta = \dots A \dots} P_G(A \Rightarrow^+ \beta | \Theta).$$

**Lemma 1** A linear PCFG  $G$  with parameters  $\Theta$  which does not have any nonterminating nonterminals is tight.

*Proof:* Our proof relies on the properties of a certain  $|N| \times |N|$  matrix  $M$  where:

$$M_{AB} = \sum_{A \rightarrow \beta \in R_A} n(\beta, B) \theta_{A \rightarrow \beta}$$

where  $n(\beta, B)$  is the number of appearances of the nonterminal  $B$  in the sequence  $\beta$ .  $M_{AB}$  is the expected number of  $B$  nonterminals generated from an  $A$  nonterminal in one single derivational step, so  $[M^k]_{AB}$  is the expected number of  $B$  nonterminals generated from an  $A$  nonterminal in a  $k$ -step derivation (Wetherell, 1980).

Since  $M$  is a non-negative matrix, under some regularity conditions, the Frobenius-Perron theorem states that the largest eigenvalue of this matrix (in absolute value) is a real number. Let this eigenvalue be denoted by  $\lambda$ .

A PCFG is called “subcritical” if  $\lambda < 1$  and supercritical if  $\lambda > 1$ . Then, in turn, a PCFG is tight if it is subcritical. It is not tight if it is supercritical. The case of  $\lambda = 1$  is a borderline case that does not give sufficient information to know whether the PCFG is tight or not. In the Bayesian case, for a continuous prior such as the Dirichlet prior, this borderline case will have measure zero under the prior.

Now let  $A \in N$ . Since the grammar is linear, there is no derivation  $A \Rightarrow^+ \dots A \dots A \dots$ . Therefore, any derivation of the form  $A \Rightarrow^+ \dots A \dots$  includes  $A$  on the right hand-side exactly once. Because the grammar has no useless nonterminals, the probability of such a derivation is strictly smaller than 1.

For each  $A \in N$ , define:

$$p_A = \sum_{\beta = \dots A \dots} P(A \Rightarrow^{|\beta|} \beta | \Theta).$$

Since  $A$  is not useless, then  $p_A < 1$ . Therefore  $q = \max_A p_A < 1$ . Since any derivation of length  $k$  of the form  $A \Rightarrow \dots A \dots$  can be decomposed to at least  $\frac{k}{2^{|N|}}$  cycles that start at a terminal  $B \in N$  and end in the same nonterminal  $B \in N$ , it holds that:

$$[M^k]_{AA} \leq q^{\frac{k}{2^{|N|}}} \xrightarrow{k \rightarrow \infty} 0.$$

This means that  $\text{trace}(M^k) \xrightarrow{k \rightarrow \infty} 0$ . This means that the eigenvalue of  $M$  is strictly smaller than 1 (linear algebra), and therefore the PCFG is tight. ■

**Proposition 1** Any continuous prior  $P(\Theta)$  on a linear grammar  $G$  is tight almost everywhere for  $G$ .

*Proof:* Let  $G$  be a linear grammar. With a continuous prior, the probability of  $G$  getting parameters from the prior which yield a useless non-terminal is 0 – it would require setting at least one rule in the grammar with rule probability which is exactly 1. Therefore, with probability 1, the parameters taken from the prior yield a PCFG which is linear and does not have nonterminating nonterminals. According to Lemma 1, this means the PCFG is tight. ■

Deciding whether a grammar  $G$  is linear can be done in polynomial time using the construction from Bar-Hillel et al. (1964). We can first eliminate the differences between nonterminals and terminal symbols by adding a rule  $A \rightarrow c_A$  for each nonterminal  $A \in N$ , after extending the set of terminal symbols  $A$  with  $\{c_A | A \in N\}$ . Let  $G_A$  be the grammar  $G$  with the start symbol being replaced with  $A$ . We can then intersect the grammar  $G_A$  with the regular language  $T^*c_A T^*c_A T^*$  (for each nonterminal  $A \in N$ ). If for any nonterminal  $A$  the intersection is not the empty set (with respect to the language that the intersection generates), then the grammar is not linear. Checking whether the intersection is the empty set or not can be done in polynomial time.

We conclude this section by remarking that many of the models used in computational linguistics are in fact equivalent to linear PCFGs, so continuous Bayesian priors are almost everywhere tight. For example, HMMs and many kinds of “stacked” finite-state machines are equivalent to

linear PCFGs, as are the example PCFGs given in Johnson et al. (2007) to motivate the MCMC estimation procedures.

## 5 Dirichlet priors

The first step in Bayesian inference is to specify a prior on  $\Theta$ . In the rest of this paper we take  $P(\Theta)$  to be a product of Dirichlet distributions, with one distribution for each non-terminal  $A \in N$ , as this turns out to simplify the computations considerably. The prior is parameterized by a positive real valued vector  $\alpha$  indexed by productions  $R$ , so each production probability  $\theta_{A \rightarrow \beta}$  has a corresponding Dirichlet parameter  $\alpha_{A \rightarrow \beta}$ . As before, let  $R_A$  be the set of productions in  $R$  with left-hand side  $A$ , and let  $\theta_A$  and  $\alpha_A$  refer to the component subvectors of  $\theta$  and  $\alpha$  respectively indexed by productions in  $R_A$ . The Dirichlet prior  $P(\Theta | \alpha)$  is:

$$P(\Theta | \alpha) = \prod_{A \in N} P_D(\Theta_A | \alpha_A),$$

where

$$P_D(\Theta_A | \alpha_A) = \frac{1}{C(\alpha_A)} \prod_{r \in R_A} \theta_r^{\alpha_r - 1} \quad \text{and}$$

$$C(\alpha_A) = \frac{\prod_{r \in R_A} \Gamma(\alpha_r)}{\Gamma(\sum_{r \in R_A} \alpha_r)}$$

where  $\Gamma$  is the generalized factorial function and  $C(\alpha)$  is a normalization constant that does not depend on  $\Theta_A$ .

Dirichlet priors are useful because they are *conjugate* to the multinomial distribution, which is the building block of PCFGs. Ignoring issues of tightness for the moment and setting  $P(t | \Theta) = \mu_\Theta(t)$ , this means that in the supervised setting the posterior distribution  $P(\Theta | \mathbf{t}, \alpha)$  given a set of parse trees  $\mathbf{t} = (t_1, \dots, t_n)$  is also a product of Dirichlets distribution.

$$P(\Theta | \mathbf{t}, \alpha) \propto P(\mathbf{t} | \Theta) P(\Theta | \alpha)$$

$$\propto \left( \prod_{r \in R} \theta_r^{f_r(\mathbf{t})} \right) \left( \prod_{r \in R} \theta_r^{\alpha_r - 1} \right)$$

$$= \prod_{r \in R} \theta_r^{f_r(\mathbf{t}) + \alpha_r - 1}$$

which is a product of Dirichlet distributions with parameters  $\mathbf{f}(\mathbf{t}) + \alpha$ , where  $\mathbf{f}(\mathbf{t})$  is the vector of rule counts in  $\mathbf{t}$  indexed by  $r \in R$ . We can thus write:

$$P(\Theta | \mathbf{t}, \alpha) = P(\Theta | \mathbf{f}(\mathbf{t}) + \alpha)$$

**Input:** Grammar  $G$ , vector of trees  $\mathbf{t}$ , vector of hyperparameters  $\alpha$ , previous parameters  $\Theta_0$ .

**Result:** A vector of parameters  $\Theta$

**repeat**

    draw  $\theta$  from products of Dirichlet with hyperparameters  $\alpha + \mathbf{f}(\mathbf{t})$

**until**  $\Theta$  is tight for  $G$ ;

return  $\Theta$

**Algorithm 1:** An algorithm for generating samples from  $P(\Theta | \mathbf{t}, \alpha)$  for the only-tight approach.

**Input:** Grammar  $G$ , vector of trees  $\mathbf{t}$ , vector of hyperparameters  $\alpha$ , previous rule parameters  $\Theta_0$ .

**Result:** A vector of parameters  $\Theta$

draw a proposal  $\Theta^*$  from a product of Dirichlets with parameters  $\alpha + \mathbf{f}(\mathbf{t})$ .

draw a uniform number  $u$  from  $[0, 1]$ .

if  $u < \min\{1, (Z(\Theta^{(i-1)})/Z(\Theta^*))^n\}$  return  $\Theta^*$ .

return  $\Theta_0$ .

**Algorithm 2:** One step of Metropolis-Hastings algorithm for generating samples from  $P(\Theta | \mathbf{t}, \alpha)$  for the renormalization approach.

which makes it clear that the rule counts are directly added to the parameters of the prior to produce the parameters of the posterior.

## 6 Inference in the supervised setting

We first discuss Bayesian inference in the supervised setting, as inference in the unsupervised setting is based on inference for the supervised setting. For each of the three approaches to non-tightness we provide an algorithm that characterizes the posterior  $P(\Theta | \mathbf{t})$ , where  $\mathbf{t} = (t_1, \dots, t_n)$  is a sequence of trees, by generating samples from that posterior. Our MCMC algorithms for the unsupervised setting build on these samplers for the supervised setting.

### 6.1 The only-tight approach

The ‘‘only-tight’’ approach requires that the prior assign zero mass to non-tight rule probability vectors  $\Theta^\perp$ . One way to define such a distribution is to restrict the domain of an existing prior distribution with the set of tight  $\Theta$  and renormalize. In more detail, if  $P(\Theta)$  is a prior over rule probabilities, then its renormalization is the prior  $P'$  defined as:

$$P'(\Theta) = \frac{P(\Theta)I(\Theta \notin \Theta^\perp)}{Z(\Theta^\perp)}. \quad (2)$$

where  $Z(\Theta^\perp) = \int_{\Theta} P(\Theta)I(\Theta \notin \Theta^\perp)d\Theta$ .

**Input:** Grammar  $G$ , vector of trees  $\mathbf{t}$ , vector of hyperparameters  $\alpha$ , previous parameters  $\Theta_0$ .

**Result:** A vector of parameters  $\Theta$   
draw  $\Theta$  from products of Dirichlet with hyperparameters  $\alpha + \mathbf{f}(\mathbf{t})$   
return  $\Theta$

**Algorithm 3:** An algorithm for generating samples from  $P(\Theta | \mathbf{t}, \alpha)$  for the sink-state approach.

Perhaps surprisingly, it turns out that if  $P(\Theta)$  belongs to a family of conjugate priors, then  $P'(\Theta)$  also belongs to a (different) family of conjugate priors as well.

**Proposition 2** *Let  $P(\Theta|\alpha)$  be a prior with hyperparameters  $\alpha$  over the parameters of  $G$  such that  $P$  is conjugate to the grammar likelihood. Then  $P'$ , defined in Eq. 2, is conjugate to the grammar likelihood as well.*

*Proof:* Assume that trees  $\mathbf{t}$  are observed, and the prior over the grammar parameters is the prior defined in Eq. 2. Therefore, the posterior is:

$$\begin{aligned} P(\Theta|\mathbf{t}, \alpha) &\propto P'(\Theta|\alpha)p(\mathbf{t}|\Theta) \\ &= \frac{P(\Theta|\alpha)p(\mathbf{t}|\Theta)I(\Theta \notin \Theta^\perp)}{Z(\Theta^\perp)} \\ &\propto \frac{P(\Theta|\mathbf{t}, \alpha)I(\Theta \notin \Theta^\perp)}{Z(\Theta^\perp)}. \end{aligned}$$

Since  $P(\Theta|\alpha)$  is a conjugate prior to the PCFG likelihood, then there exists  $\alpha' = \alpha'(\mathbf{t})$  such that  $P(\Theta|\mathbf{t}, \alpha) = P'(\Theta|\alpha')$ . Therefore:

$$P(\Theta|\mathbf{t}, \alpha) \propto \frac{P(\Theta|\alpha')I(\Theta \notin \Theta^\perp)}{Z(\Theta^\perp)}.$$

which exactly equals  $P'(\Theta|\alpha')$ . ■

Sampling from the posterior over the parameters given a set of trees  $\mathbf{t}$  is therefore quite simple when assuming the base prior being renormalized is a product of Dirichlets. Algorithm 1 samples from a product of Dirichlets distribution with hyperparameters  $\alpha + \mathbf{f}(\mathbf{t})$  repeatedly, each time checking and rejecting the sample until we obtain a tight PCFG.

The more mass the Dirichlet distribution with hyperparameters  $\alpha + \mathbf{f}(\mathbf{t})$  puts on non-tight PCFGs, the more rejections will happen. In general, if the probability mass on non-tight PCFGs is  $q_\perp$ , then it would require, on average  $1/(1 - q_\perp)$  samples from this distribution in order to obtain a tight PCFG.

## 6.2 The renormalization approach

The renormalization approach modifies the likelihood function instead of the prior. Here we use a product of Dirichlets prior  $P(\Theta | \alpha)$  on rule probability vectors  $\Theta$ , but the presence of the partition function  $Z(\Theta)$  in Eq. 1 means that the likelihood is no longer conjugate to the prior. Instead we have:

$$\begin{aligned} P(\Theta | \mathbf{t}) &= \prod_{i=1}^n \frac{\mu_\Theta(t_i)}{Z(\Theta)} P(\Theta | \alpha) \\ &\propto \frac{1}{Z(\Theta)^n} P(\Theta | \alpha + \mathbf{f}(\mathbf{t})). \end{aligned} \quad (3)$$

Note that the factor  $Z(\Theta)$  depends on  $\Theta$ , and therefore cannot be absorbed into the constant. Algorithm 2 describes a Metropolis-Hastings sampler for sampling from the posterior in Eq. 3 that uses a product of Dirichlets with parameters  $\alpha + \mathbf{f}(\mathbf{t})$  as a proposal distribution.

In our experiments, we use the algorithm from Nederhof and Satta (2008) to compute the partition function which is needed in Algorithm 2.

## 6.3 The “sink element” approach

The “sink element” approach does not affect the likelihood (since the probability of a tree  $t$  is just the product of the probabilities of the rules used to generate it), nor does it require a change to the prior. (The sink element  $\perp$  is not a member of the set of trees  $\mathcal{T}$ , so it cannot appear in the data  $\mathbf{t}$ ).

This means that the conjugacy argument given at the bottom of section 5 holds in this approach, so the posterior  $P(\Theta | \mathbf{t}, \alpha)$  is a product of Dirichlets with parameters  $\mathbf{f}(\mathbf{t}) + \alpha$ . Algorithm 3 gives a sampler for  $P(\Theta | \mathbf{t}, \alpha)$  for the sink element approach.

## 7 Inference in the unsupervised setting

Johnson et al. (2007) provide two Markov chain Monte Carlo algorithms for Bayesian inference for PCFG rule probabilities in the unsupervised setting (i.e., where the data consists of a corpus of strings  $\mathbf{w} = (w_1, \dots, w_n)$  alone). The algorithms we give here are based on their Gibbs sampler, which in each iteration first samples parse trees  $\mathbf{t} = (t_1, \dots, t_n)$ , where each  $t_i$  is a parse for  $w_i$ , from  $P(\mathbf{t} | \mathbf{w}, \Theta)$ , and then samples  $\Theta$  from  $P(\Theta | \mathbf{t}, \alpha)$ .

Notice that the conditional distribution  $P(t | \mathbf{w}, \Theta)$  is unaffected in each of our three approaches (the partition functions cancel in the

**Input:** Grammar  $G$ , vector of hyperparameters  $\alpha$ , vector of strings  $\mathbf{w} = (w_1, \dots, w_n)$ , previous rule parameters  $\Theta_0$ .

**Result:** A vector of parameters  $\Theta$

**for**  $i \leftarrow 1$  **to**  $n$  **do**

    | draw  $t_i$  from  $P(t_i|w_i, \Theta_0)$

**end**

use Algorithm 2 to sample  $\Theta$  given  $G, \mathbf{t}, \alpha$  and  $\Theta_0$

return  $\Theta$

**Algorithm 4:** One step of the Metropolis-within-Gibbs sampler for the renormalization approach.

renormalization approach), so the algorithm for sampling from  $P(\mathbf{t} \mid \mathbf{w}, \Theta)$  given by Johnson et al. applies in each of our three approaches as well.

Johnson et al. ignored tightness and assumed that  $P(\Theta \mid \mathbf{t}, \alpha)$  is a product of Dirichlets with parameters  $\mathbf{f}(\mathbf{t}) + \alpha$ . As we noted in section 6.3, this assumption holds for the sink-state approach to non-tightness, so their sampler is in fact correct for the sink-state approach.

In fact, we obtain samplers for the unsupervised setting for each of our approaches by “plugging in” the corresponding sampling algorithm (Eq. 1–3) for  $P(\Theta \mid \mathbf{t}, \alpha)$  into the generic Gibbs sampler framework of Johnson et al.

The one complication is that because we use a Metropolis-Hastings procedure to generate samples from  $P(\Theta \mid \mathbf{t}, \alpha)$  in the renormalization approach, we use the Metropolis-within-Gibbs procedure given in Algorithm 4 (Robert and Casella, 2004).

## 8 The expressive power of the three approaches

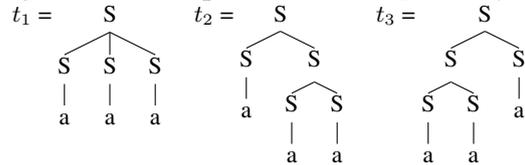
Probably the most important question to ask with respect to the three different approaches to non-tightness is whether they differ in terms of expressive power. Clearly the three approaches differ in terms of the grammars they admit (the only-tight approach requires the prior to only assign non-zero probability to tight PCFGs, while the other two approaches permit the prior to assign non-zero probability to non-tight PCFGs as well). However, if we regard a grammar as merely a device for defining a distribution over trees and a prior as defining a distribution over distributions over trees, it is reasonable to ask whether the class of distributions over distributions of trees that each of these approaches define are the same or differ. We believe, but have not proved, that all three approaches define the same class of distributions over distribu-

tions of trees in the following sense: any prior used with one of the approaches can be transformed into a different prior that can be used with one of the other approaches, and yield the same posterior over trees conditioned on a string, marginalizing out the parameters.

This does not mean that the three approaches are equivalent, however. In this section we provide a grammar such that with a uniform prior over rule probabilities, the conditional distribution over trees given a fixed string varies under each of the three different approaches.

The grammar we consider has three rules  $S \rightarrow S S S \mid S S \mid a$  with probabilities  $\theta_1, \theta_2$  and  $1 - \theta_1 - \theta_2$ , respectively. The  $\Theta$  parameters are required to satisfy  $\theta_1 + \theta_2 \leq 1$  and  $\theta_i \geq 0$  for  $i = 1, 2$ .

We compute the posterior distribution over parse trees for the string  $w = a a a$ . The grammar generates three parse trees for  $w_1$ , namely:



The partition function  $Z$  for this grammar is the smallest positive root of the cubic equation:

$$Z = \theta_1 Z^3 + \theta_2 Z^2 + (1 - \theta_1 - \theta_2)$$

We used Mathematica to find an analytic solution for  $Z$  in this equation, obtaining not only an expression for the partition function  $Z(\Theta)$  but also identifying the non-tight region  $\Theta^\perp$ .

In order to compute  $P(t_i|w)$ , we used Mathematica to first compute the following quantities:

$$q_{\text{sinkElement}}(t_i) = \int_{\Theta} \mu_{\Theta}(t_i) d\Theta$$

$$q_{\text{tightOnly}}(t_i) = \int_{\Theta} \mu_{\Theta}(t_i) I(\Theta \notin \Theta^\perp) d\Theta$$

$$q_{\text{renormalization}}(t_i) = \int_{\Theta} \mu_{\Theta}(t_i) / Z(\Theta) d\Theta$$

where  $i \in \{1, 2, 3\}$ . We used Mathematica to analytically compute  $q(t_i)$  for each approach and each  $i \in \{1, 2, 3\}$ . Then it’s easy to show that:

$$P(t_i \mid w) = \frac{q(t_i)}{\sum_{i'=1}^3 q(t_{i'})}$$

where the  $q$  used is based on the approach to tightness desired. For the sink-element approach,

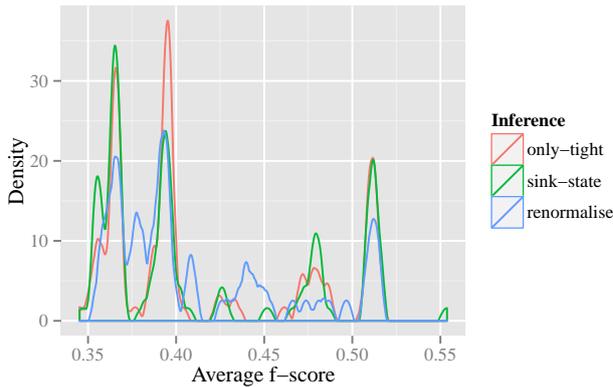


Figure 1: The density of the  $F_1$ -scores with the three approaches. The prior used is a symmetric Dirichlet with  $\alpha = 0.1$ .

$P(t_1|w) = \frac{7}{11} \approx 0.636364$ . For the only-tight approach  $P(t_1|w) = \frac{11179}{17221} \approx 0.649149$ . For the renormalization approach the analytic expression is too complex to include in this paper, but it approximately equals 0.619893. A log of our Mathematica calculations is available at <http://www.cs.columbia.edu/~scohen/acl13tightness-mathematica.pdf>, and we confirmed these results to three decimal places using the samplers described above (which required  $10^7$  samples per approach).

While the differences between these conditional probabilities are not great, the conditional probabilities are clearly different, so the three approaches do in fact define different distributions over trees under a uniform prior on rule probabilities.

## 9 Empirical effects of the three approaches in unsupervised grammar induction

In this section we present experiments using the three samplers just described in an unsupervised grammar induction problem. Our goal here is not to improve the state-of-the-art in unsupervised grammar induction, but to try to measure empirical differences in the estimates produced by the three different approaches to tightness just described. The bottom line of our experiments is that we could not detect any significant difference in the estimates produced by samplers for these three different approaches.

In our experiments we used the English Penn treebank (Marcus et al., 1993). We use the part-

of-speech tag sequences of sentences shorter than 11 words in sections 2–21. The grammar we use is the PCFG version of the dependency model with valence (Klein and Manning, 2004), as it appears in Smith (2006).

We used a symmetric Dirichlet prior with hyperparameter  $\alpha = 0.1$ . For each of the three approaches for handling tightness, we ran 100 times the samplers in §7, each for 1,000 iterations. We discarded the first 900 sweeps of each run, and calculated the  $F_1$ -scores of the sampled trees every 10th sweep from the last 100 sweeps. For each run we calculated the average  $F_1$ -score over the 10 sweeps we evaluated. We thus have 100 average  $F_1$ -scores for each of the samplers.

Figure 1 plots the density of  $F_1$  scores (compared to the gold standard) resulting from the Gibbs sampler, using all three approaches. The mean value for each of the approaches is 0.41 with standard deviation 0.06 (only-tight), 0.41 with standard deviation 0.05 (renormalization) and 0.42 with standard deviation 0.06 (sink element). In addition, the only-tight approach results in an average of 437 (s.d., 142) rejected proposals in 1,000 samples, while the renormalization approach results in an average of 232 (s.d., 114) rejected proposals in 1,000 samples. (It’s not surprising that the only-tight approach results in more rejections as it keeps proposing new  $\Theta$  until a tight proposal is found, while the renormalization approach simply uses the old  $\Theta$ ).

We performed two-sample Kolmogorov-Smirnov tests (which are non-parametric tests designed to determine if two distributions are different; see DeGroot, 1991) on each of the three pairs of 100  $F_1$ -scores. None of the tests were close to significant; the p-values were all above 0.5. Thus our experiments provided no evidence that the samplers produced different distributions over trees, although it’s reasonable to expect that these distributions do indeed differ.

In terms of running time, our implementation of the renormalization approach was several times slower than our implementations of the other two approaches because we used the naive fixed-point algorithm to compute the partition function: perhaps this could be improved using one of the more sophisticated partition function algorithms described in Nederhof and Satta (2008).

## 10 Conclusion

In this paper we characterized the notion of an almost everywhere tight grammar in the Bayesian setting and showed it holds for linear CFGs. For non-linear CFGs, we described three different approaches to handle non-tightness. The “only-tight” approach restricts attention to tight PCFGs, and perhaps surprisingly, we showed that conjugacy still obtains when the domain of a product of Dirichlets prior is restricted to the subset of tight grammars. The renormalization approach involves renormalizing the PCFG measure  $\mu$  over trees when the grammar is non-tight, which destroys conjugacy with a product of Dirichlets prior. Perhaps most surprisingly of all, the sink-element approach, which assigns the missing mass in non-tight PCFG to a sink element  $\perp$ , turns out to be equivalent to existing practice where tightness is ignored.

We studied the posterior distributions over trees induced by the three approaches under a uniform prior for a simple grammar and showed that they differ. We leave for future work the important question of whether the classes of distributions over distributions over trees that the three approaches define are the same or different.

We described samplers for the supervised and unsupervised settings for each of these approaches, and applied them to an unsupervised grammar induction problem. (The code for the unsupervised samplers is available from <http://web.science.mq.edu.au/~mjohnson>).

We could not detect any difference in the posterior distributions over trees produced by these samplers, despite devoting considerable computational resources to the problem. This suggests that for these kinds of problems at least, tightness is not of practical concern for Bayesian inference of PCFGs.

## Acknowledgements

We thank the anonymous reviewers and Giorgio Satta for their valuable comments. Shay Cohen was supported by the National Science Foundation under Grant #1136996 to the Computing Research Association for the CIFellows Project, and Mark Johnson was supported by the Australian Research Council’s Discovery Projects funding scheme (project numbers DP110102506 and DP110102593).

## References

- K. B. Atherya and P. E. Ney. 1972. *Branching Processes*. Dover Publications.
- Y. Bar-Hillel, M. Perles, and E. Shamir. 1964. On formal properties of simple phrase structure grammars. *Language and Information: Selected Essays on Their Theory and Application*, pages 116–150.
- T. L. Booth and R. A. Thompson. 1973. Applying probability measures to abstract languages. *IEEE Transactions on Computers*, C-22:442–450.
- Z. Chi and S. Geman. 1998. Estimation of probabilistic context-free grammars. *Computational Linguistics*, 24(2):299–305.
- Z. Chi. 1999. Statistical properties of probabilistic context-free grammars. *Computational Linguistics*, 25(1):131–160.
- S. B. Cohen and N. A. Smith. 2012. Empirical risk minimization for probabilistic grammars: Sample complexity and hardness of learning. *Computational Linguistics*, 38(3):479–526.
- M. H. DeGroot. 1991. *Probability and Statistics (3rd edition)*. Addison-Wesley.
- M. Johnson, T. L. Griffiths, and S. Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proceedings of NAACL*.
- D. Klein and C. D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of ACL*.
- K. Kurihara and T. Sato. 2006. Variational Bayesian grammar induction for natural language. In *8th International Colloquium on Grammatical Inference*.
- K. Lari and S.J. Young. 1990. The estimation of Stochastic Context-Free Grammars using the Inside-Outside algorithm. *Computer Speech and Language*, 4(35-56).
- M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19:313–330.
- M.-J. Nederhof and G. Satta. 2008. Computing partition functions of PCFGs. *Research on Language and Computation*, 6(2):139–162.
- C. P. Robert and G. Casella. 2004. *Monte Carlo Statistical Methods*. Springer-Verlag New York.
- N. A. Smith. 2006. *Novel Estimation Methods for Unsupervised Discovery of Latent Structure in Natural Language Text*. Ph.D. thesis, Johns Hopkins University.
- C. S. Wetherell. 1980. Probabilistic languages: A review and some open questions. *Computing Surveys*, 12:361–379.

# Integrating Multiple Dependency Corpora for Inducing Wide-coverage Japanese CCG Resources

Sumire Uematsu†

uematsu@cks.u-tokyo.ac.jp

Takuya Matsuzaki‡

takuya-matsuzaki@nii.ac.jp

Hiroki Hanaoka†

hanaoka@nii.ac.jp

Yusuke Miyao†

yusuke@nii.ac.jp

Hideki Mima†

mima@t-adm.t.u-tokyo.ac.jp

†The University of Tokyo

Hongo 7-3-1, Bunkyo, Tokyo, Japan

‡National Institute of Informatics

Hitotsubashi 2-1-2, Chiyoda, Tokyo, Japan

## Abstract

This paper describes a method of inducing wide-coverage CCG resources for Japanese. While deep parsers with corpus-induced grammars have been emerging for some languages, those for Japanese have not been widely studied, mainly because most Japanese syntactic resources are dependency-based. Our method first integrates multiple dependency-based corpora into phrase structure trees and then converts the trees into CCG derivations. The method is empirically evaluated in terms of the coverage of the obtained lexicon and the accuracy of parsing.

## 1 Introduction

Syntactic parsing for Japanese has been dominated by a dependency-based pipeline in which chunk-based dependency parsing is applied and then semantic role labeling is performed on the dependencies (Sasano and Kurohashi, 2011; Kawahara and Kurohashi, 2011; Kudo and Matsumoto, 2002; Iida and Poesio, 2011; Hayashibe et al., 2011). This dominance is mainly because chunk-based dependency analysis looks most appropriate for Japanese syntax due to its morphosyntactic typology, which includes agglutination and scrambling (Bekki, 2010). However, it is also true that this type of analysis has prevented us from deeper syntactic analysis such as deep parsing (Clark and Curran, 2007) and logical inference (Bos et al., 2004; Bos, 2007), both of which have been surpassing shallow parsing-based approaches in languages like English.

In this paper, we present our work on inducing wide-coverage Japanese resources based on

combinatory categorial grammar (CCG) (Steedman, 2001). Our work is basically an extension of a seminal work on CCGbank (Hockenmaier and Steedman, 2007), in which the phrase structure trees of the Penn Treebank (PTB) (Marcus et al., 1993) are converted into CCG derivations and a wide-coverage CCG lexicon is then extracted from these derivations. As CCGbank has enabled a variety of outstanding works on wide-coverage deep parsing for English, our resources are expected to significantly contribute to Japanese deep parsing.

The application of the CCGbank method to Japanese is not trivial, as resources like PTB are not available in Japanese. The widely used resources for parsing research are the Kyoto corpus (Kawahara et al., 2002) and the NAIST text corpus (Iida et al., 2007), both of which are based on the dependency structures of chunks. Moreover, the relation between chunk-based dependency structures and CCG derivations is not obvious.

In this work, we propose a method to integrate multiple dependency-based corpora into phrase structure trees augmented with predicate argument relations. We can then convert the phrase structure trees into CCG derivations. In the following, we describe the details of the integration method as well as Japanese-specific issues in the conversion into CCG derivations. The method is empirically evaluated in terms of the quality of the corpus conversion, the coverage of the obtained lexicon, and the accuracy of parsing with the obtained grammar. Additionally, we discuss problems that remain in Japanese resources from the viewpoint of developing CCG derivations.

There are three primary contributions of this paper: 1) we show the first comprehensive results for Japanese CCG parsing, 2) we present a methodology for integrating multiple dependency-based re-

$I$	$\frac{give}{S \backslash NP / NP / NP :}$	$\frac{them}{NP : them'}$	$\frac{money}{NP : money'}$	
$NP : I'$	$\frac{\lambda x \lambda y \lambda z. give' y x z}{S \backslash NP / NP : \lambda y \lambda z. give' y them' z} >$			
	$\frac{S \backslash NP : \lambda z. give' money' them' z}{S : give' money' them' I'} <$			$>$

Figure 1: A CCG derivation.

$$\begin{aligned}
 X/Y : f \quad Y : a &\rightarrow X : fa && (>) \\
 Y : a \quad X \backslash Y : a &\rightarrow X : fa && (<) \\
 X/Y : f \quad Y/Z : g &\rightarrow X/Z : \lambda x. f(gx) && (> B) \\
 Y \backslash Z : g \quad X \backslash Y : f &\rightarrow X \backslash Z : \lambda x. f(gx) && (< B)
 \end{aligned}$$

Figure 2: Combinatory rules (used in the current implementation).

sources to induce CCG derivations, and 3) we investigate the possibility of further improving CCG analysis by additional resources.

## 2 Background

### 2.1 Combinatory Categorial Grammar

CCG is a syntactic theory widely accepted in the NLP field. A grammar based on CCG theory consists of *categories*, which represent syntactic categories of words and phrases, and *combinatory rules*, which are rules to combine the categories. Categories are either *ground categories* like  $S$  and  $NP$  or *complex categories* in the form of  $X/Y$  or  $X \backslash Y$ , where  $X$  and  $Y$  are the categories. Category  $X/Y$  intuitively means that it becomes category  $X$  when it is combined with another category  $Y$  to its right, and  $X \backslash Y$  means it takes a category  $Y$  to its left. Categories are combined by applying combinatory rules (Fig. 2) to form categories for larger phrases. Figure 1 shows a CCG analysis of a simple English sentence, which is called a *derivation*. The verb *give* is assigned category  $S \backslash NP / NP / NP$ , which indicates that it takes two NPs to its right, one NP to its left, and finally becomes  $S$ . Starting from lexical categories assigned to words, we can obtain categories for phrases by applying the rules recursively.

An important property of CCG is a clear interface between syntax and semantics. As shown in Fig. 1, each category is associated with a lambda term of semantic representations, and each combinatory rule is associated with rules for semantic composition. Since these rules are universal, we can obtain different semantic representations by switching the semantic representations of lexical categories. This means that we can plug in a vari-

Sentence	$S$	Verb	$S \backslash \$$ (e.g. $S \backslash NP_{ga}$ )
Noun phrase	$NP$	Post particle	$NP_{ga o ni to} \backslash NP$
Auxiliary verb	$S \backslash S$		

Table 1: Typical categories for Japanese syntax.

Cat.	Feature	Value	Interpretation
NP	case	ga	nominal
		o	accusative
		ni	dative
		to	comitative, complementizer, etc.
		nc	none
		vo_s	causative
S	form	stem	stem
		base	base
		neg	imperfect or negative
		cont	continuative
		vo_s	causative

Table 2: Features for Japanese syntax (those used in the examples in this paper).

ety of semantic theories with CCG-based syntactic parsing (Bos et al., 2004).

### 2.2 CCG-based syntactic theory for Japanese

Bekki (2010) proposed a comprehensive theory for Japanese syntax based on CCG. While the theory is based on Steedman (2001), it provides concrete explanations for a variety of constructions of Japanese, such as agglutination, scrambling, long-distance dependencies, etc. (Fig. 3).

The ground categories in his theory are  $S$ ,  $NP$ , and  $CONJ$  (for conjunctions). Table 1 presents typical lexical categories. While most of them are obvious from the theory of CCG, categories for auxiliary verbs require an explanation. In Japanese, auxiliary verbs are extensively used to express various semantic information, such as tense and modality. They agglutinate to the main verb in a sequential order. This is explained in Bekki’s theory by the category  $S \backslash S$  combined with a main verb via the function composition rule ( $<B$ ). Syntactic features are assigned to categories  $NP$  and  $S$  (Table 2). The feature *case* represents a syntactic case of a noun phrase. The feature *form* denotes an inflection form, and is necessary for constraining the grammaticality of agglutination.

Our implementation of the grammar basically follows Bekki (2010)’s theory. However, as a first step in implementing a wide-coverage Japanese parser, we focused on the frequent syntactic constructions that are necessary for computing predicate argument relations, including agglutination, inflection, scrambling, case alternation, etc. Other details of the theory are largely simplified (Fig. 3),



and Bender, 2002) is a large-scale Japanese grammar based on HPSG, but its semantics is tightly embedded in the grammar and it is not as easy to systematically switch them as it is in CCG. Yoshida (2005) proposed methods for extracting a wide-coverage lexicon based on HPSG from a phrase structure treebank of Japanese. We largely extended their work by exploiting the standard chunk-based Japanese corpora and demonstrated the first results for Japanese deep parsing with grammar induced from large corpora.

Corpus-based acquisition of wide-coverage CCG resources has enjoyed great success for English (Hockenmaier and Steedman, 2007). In that method, PTB was converted into CCG-based derivations from which a wide-coverage CCG lexicon was extracted. CCGbank has been used for the development of wide-coverage CCG parsers (Clark and Curran, 2007). The same methodology has been applied to German (Hockenmaier, 2006), Italian (Bos et al., 2009), and Turkish (Çakıcı, 2005). Their treebanks are annotated with dependencies of *words*, the conversion of which into phrase structures is not a big concern. A notable contribution of the present work is a method for inducing CCG grammars from chunk-based dependency structures, which is not obvious, as we discuss later in this paper.

CCG parsing provides not only predicate argument relations but also CCG derivations, which can be used for various semantic processing tasks (Bos et al., 2004; Bos, 2007). Our work constitutes a starting point for such deep linguistic processing for languages like Japanese.

### 3 Corpus integration and conversion

For wide-coverage CCG parsing, we need a) a wide-coverage CCG lexicon, b) combinatory rules, c) training data for parse disambiguation, and d) a parser (e.g., a CKY parser). Since d) is grammar- and language-independent, all we have to develop for a new language is a)–c).

As we have adopted the method of CCGbank, which relies on a source treebank to be converted into CCG derivations, a critical issue to address is the absence of a Japanese counterpart to PTB. We only have chunk-based dependency corpora, and their relationship to CCG analysis is not clear.

Our solution is to first integrate multiple dependency-based resources and convert them into a phrase structure treebank that is independent

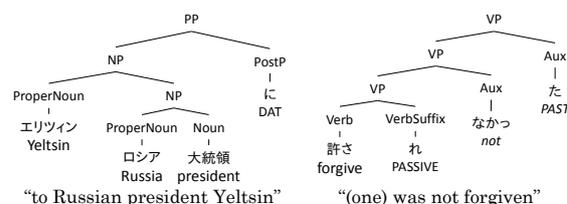


Figure 6: Internal structures of a nominal chunk (left) and a verbal chunk (right).

of CCG analysis (Step 1). Next, we translate the treebank into CCG derivations (Step 2). The idea of Step 2 is similar to what has been done with the English CCGbank, but obviously we have to address language-specific issues.

#### 3.1 Dependencies to phrase structure trees

We first integrate and convert available Japanese corpora—namely, the Kyoto corpus, the NAIST text corpus, and the JP corpus—into a phrase structure treebank, which is similar in spirit to PTB. Our approach is to convert the dependency structures of the Kyoto corpus into phrase structures and then augment them with syntactic/semantic roles from the other two corpora.

The conversion involves two steps: 1) recognizing the chunk-internal structures, and 2) converting inter-chunk dependencies into phrase structures. For 1), we don’t have any explicit information in the Kyoto corpus although, in principle, each chunk has internal structures (Vadas and Curran, 2007; Yamada et al., 2010). The lack of a chunk-internal structure makes the dependency-to-constituency conversion more complex than a similar procedure by Bos et al. (2009) that converts an Italian dependency treebank into constituency trees since their dependency trees are annotated down to the level of each word. For the current implementation, we abandon the idea of identifying exact structures and instead basically rely on the following generic rules (Fig. 6):

**Nominal chunks** Compound nouns are first formed as a right-branching phrase and post-positions are then attached to it.

**Verbal chunks** Verbal chunks are analyzed as left-branching structures.

The rules amount to assume that all but the last word in a compound noun modify the head noun (i.e., the last word) and that a verbal chunk is typically in a form  $V A_1 \dots A_n$ , where  $V$  is a verb

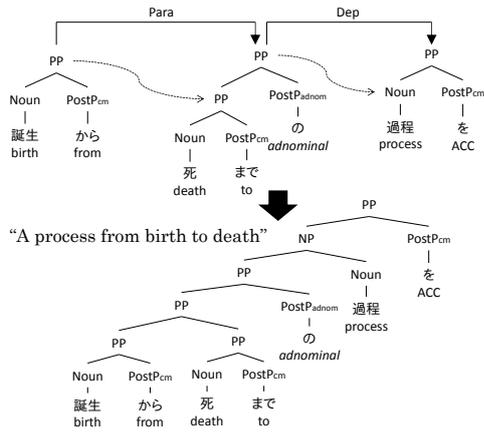


Figure 7: From inter-chunk dependencies to a tree.

(or other predicative word) and  $A_i$ s are auxiliaries (see Fig. 6). We chose the left-branching structure as default for verbal chunks because the semantic scopes of the auxiliaries are generally in that order (i.e.,  $A_1$  has the narrowest scope). For both cases, phrase symbols are percolated upward from the right-most daughters of the branches (except for a few cases like punctuation) because in almost all cases the syntactic head of a Japanese phrase is the right-most element.

In practice, we have found several patterns of exceptions for the above rules. We implemented exceptional patterns as a small CFG and determined the chunk-internal structures by deterministic parsing with the generic rules and the CFG. For example, two of the rules we came up with are

- rule A: Number  $\rightarrow$  PrefixOfNumber Number
- rule B: ClassifierPhrase  $\rightarrow$  Number Classifier

in the precedence: rule A  $>$  B  $>$  generic rules. Using the above, we bracket a compound noun

約            千            人            死亡  
*approximately   thousand   people   death*  
 PrefixOfNumber   Number   Classifier   CommonNoun  
 “death of approximately one thousand people”

as in

(((約            千)            人)            死亡)  
 (((approximately   thousand)   people)   death)

We can improve chunk-internal structures to some extent by refining the CFG rules. A complete solution like the manual annotation by Vadas and Curran (2007) is left for future work.

The conversion of inter-chunk dependencies into phrase structures may sound trivial, but it is not necessarily easy when combined with chunk-internal structures. The problem is to which node in the internal structure of the head the dependent

dep	modifier-type	precedence
Para	から/PostP <sub>cm</sub>	まで/PostP <sub>cm</sub> , */(Verb Aux), ...
Dep	*/PostP <sub>cm</sub>	*/(Verb Aux), */Noun, ...
Dep	*/PostP <sub>adnom</sub>	*/Noun, */(Verb Aux), ...

Table 3: Rules to determine adjoin position.

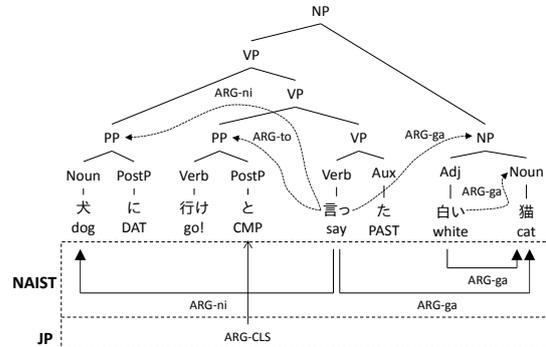


Figure 8: Overlay of pred-arg structure annotation (“The white cat who said “Go!” to the dog.”).

tree is adjoined (Fig. 7). In the case shown in the figure, three chunks are in the dependency relation indicated by arrows on the top. The dotted arrows show the nodes to which the subtrees are adjoined.

Without any human-created resources, we cannot always determine the adjoin positions correctly. Therefore, as a compromise, we wound up implementing approximate heuristic rules to determine the adjoin positions. Table 3 shows examples of such rules. A rule specifies a precedence of the possible adjoin nodes as an ordered list of patterns on the lexical head of the subtree under an adjoin position. The precedence is defined for each combination of the type of the dependent phrase, which is determined by its lexical head, and the dependency type in the Kyoto corpus.

To select the adjoin position for the left-most subtree in Fig. 7, for instance, we look up the rule table using the dependency type, “Para”, and the lexical head of the modifier subtree, “から/PostP<sub>cm</sub>”, as the key, and find the precedence “まで/PostP<sub>cm</sub>, \*/(Verb|Aux), ...”. We thus select the PP-node on the middle subtree indicated by the dotted arrow because its lexical head (the right-most word), “まで/PostP<sub>cm</sub>”, matches the first pattern in the precedence list. In general, we seek for an adjoin node for each pattern  $p$  in the precedence list, until we find a first match.

The semantic annotation given in the NAIST corpus and the JP corpus is overlaid on the phrase structure trees with slight modifications (Fig. 8).

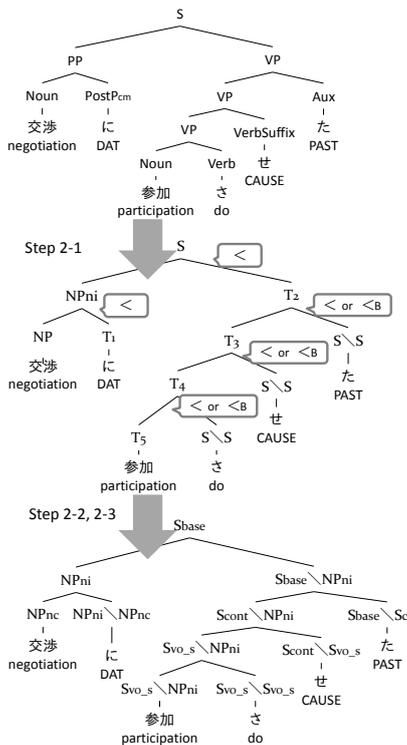


Figure 9: A phrase structure into a CCG derivation.

In the figure, the annotation given in the two corpora is shown inside the dotted box at the bottom. We converted the predicate-argument annotations given as labeled word-to-word dependencies into the relations between the predicate words and their argument *phrases*. The results are thus similar to the annotation style of PropBank (Palmer et al., 2005). In the NAIST corpus, each pred-arg relation is labeled with the argument-type (*ga/o/ni*) and a flag indicating that the relation is mediated by either a syntactic dependency or a zero anaphora. For a relation of a predicate  $w_p$  and its argument  $w_a$  in the NAIST corpus, the boundary of the argument phrase is determined as follows:

1. If  $w_a$  precedes  $w_p$  and the relation is mediated by a syntactic dep., select the maximum PP that is formed by attaching one or more postpositions to the NP headed by  $w_a$ .
2. If  $w_p$  precedes  $w_a$  or the relation is mediated by a zero anaphora, select the maximum NP headed by  $w_a$  that does not include  $w_p$ .

In the figure, “犬/dog に/DAT” is marked as the ni-argument of the predicate “言つ/say” (Case 1), and “白い/white 猫/cat” is marked as its *ga*-argument (Case 2). Case 1 is for the most basic construction, where an argument PP precedes its predicate. Case

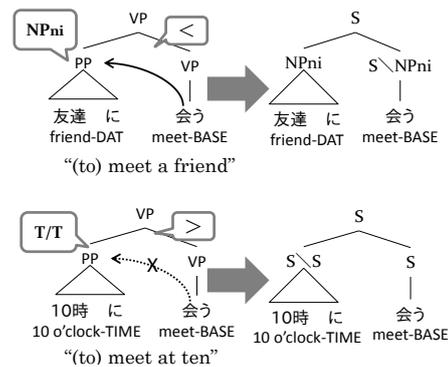


Figure 10: An argument post particle phrase (PP) (upper) and an adjunct PP (lower).

2 covers the relative clause construction, where a relative clause precedes the head NP, the modification of a noun by an adjective, and the relations mediated by zero anaphora.

The JP corpus provides only the function label to each particle “to” in the text. We determined the argument phrases marked by the “to” particles labeled as (nominal or clausal) argument-markers in a similar way to Case 1 above and identified the predicate words as the lexical heads of the phrases to which the  $PP_{to}$  phrases attach.

### 3.2 Phrase structures to CCG derivations

This step consists of three procedures (Fig. 9):

1. Add constraints on categories and features to tree nodes as far as possible and assign a combinatory rule to each branching.
2. Apply combinatory rules to all branching and obtain CCG derivations.
3. Add feature constraints to terminal nodes.

#### 3.2.1 Local constraint on derivations

According to the phrase structures, the first procedure in Step 2 imposes restrictions on the resulting CCG derivations. To describe the restrictions, we focus on some of the notable constructions and illustrate the restrictions for each of them.

**Phrases headed by case marker particles** A phrase of this type must be either an argument (Fig. 10, upper) or a modifier (Fig. 10, lower) of a predicative. Distinction between the two is made based on the pred-arg annotation of the predicative. If a phrase is found to be an argument, 1) category NP is assigned to the corresponding node, 2) the case feature of the category is given according to the particle (in the case of Fig. 10 (upper)),

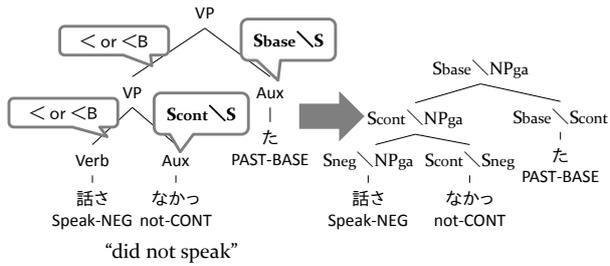


Figure 11: An auxiliary verb and its conversion.

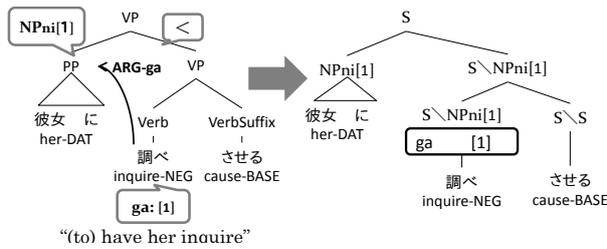


Figure 12: A causative construction.

*ni* for dative), and 3) the combinatory rule that combines the particle phrase and the predicative phrase is assigned *backward function application rule* (<). Otherwise, a category  $T/T$  is assigned to the corresponding modifier node and the rule will be *forward function application* (>).

**Auxiliary verbs** As described in Sec. 2.2, an auxiliary verb is always given the category  $S\backslash S$  and is combined with a verbal phrase via < or <B (Fig. 11). Furthermore, we assign the *form* feature value of the returning category  $S$  according to the inflection form of the auxiliary. In the case shown in the figure,  $S_{base}\backslash S$  is assigned for “た/PAST-BASE” and  $S_{cont}\backslash S$  for “なかつ/not-CONT”. As a result of this restriction, we can obtain conditions for every auxiliary agglutination because the two *form* values in  $S\backslash S$  are both restricted after applying combinatory rules (Sec. 3.2.2).

**Case alternations** In addition to the argument/adjunct distinction illustrated above, a process is needed for argument phrases of predicates involving case alternation. Such predicates are either causative (see Fig. 12) or passive verbs and can be detected by voice annotation from the NAIST corpus. For an argument of that type of verb, its *deep* case (*ga* for Fig. 12) must be used to construct the semantic representation, namely the PAS. As well as assigning the shallow case value (*ni* in Fig. 12) to the argument’s category NP, as usual, we assign a restriction to the PAS

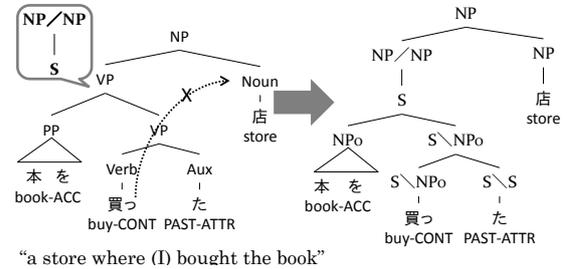
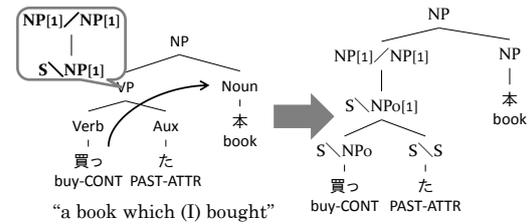


Figure 13: A relative clause with/without argument extraction (upper/lower, respectively).

of the verb so that the semantic argument corresponding to the deep case is co-indexed with the argument NP. These restrictions are then utilized for PAS construction in Sec. 3.2.3.

**Relative clauses** A relative clause can be detected as a subtree that has a VP as its left child and an NP as its right child, as shown in Fig. 13. The conversion of the subtree consists of 1) inserting a node on the top of the left VP (see the right-hand side of Fig. 13), and 2) assigning the appropriate unary rule to make the new node. The difference between candidate rules RelExt and RelIn (see Fig. 4) is whether the right-hand NP is an obligatory argument of the VP or not, which can be determined by the pred-arg annotation on the predicate in the VP. In the upper example in Fig. 13, RelIn is assigned because the right NP “book” is annotated as an accusative argument of the predicate “buy”. In contrast, RelExt is assigned in the lower side in the figure because the right NP “store” is not annotated as an argument.

**Continuous clauses** A continuous clause can be detected as a subtree with a VP of continuous form as its left child and a VP as its right child. Its conversion is similar to that of a relative clause, and only differs in that the candidate rules are Con and ConCoord. ConCoord generates a continuous clause that shares arguments with the main clause while Con produces one without shared arguments. Rule assignment is done by comparing the pred-arg annotations of the two phrases.

	Training	Develop.	Test
#Sentences	24,283	4,833	9,284
#Chunks	234,685	47,571	89,874
#Words	664,898	136,585	255,624

Table 4: Statistics of input linguistic resources.

### 3.2.2 Inverse application of rules

The second procedure in Step 2 begins with assigning a category  $S$  to the root node. A combinatorial rule assigned to each branching is then “inversely” applied so that the constraint assigned to the parent transfers to the children.

### 3.2.3 Constraints on terminal nodes

The final process consists of a) imposing restrictions on the terminal category in order to instantiate all the feature values, and b) constructing a PAS for each verbal terminal. An example of process a) includes setting the *form* features in the verb category, such as  $S \backslash NP_{ni}$ , according to the inflection form of the verb. As for b), arguments in a PAS are given according to the category and the partial restriction. For instance, if a category  $S \backslash NP_{ni}$  is obtained for “調べ/inquire” (Fig. 12), the PAS for “inquire” is unary because the category has one argument category ( $NP_{ni}$ ), and the category is co-indexed with the semantic argument *ga* in the PAS due to the partial restriction depicted in Sec. 3.2.1. As a result, a lexical entry is obtained as 調べ  $\vdash S \backslash NP_{ni}[1]: inquire([1])$ .

### 3.3 Lexical entries

Finally, lexical rules are applied to each of the obtained lexical entries in order to reduce them to the canonical form. Since words in the corpus (especially verbs) often involve pro-drop and scrambling, there are a lot of obtained entries that have slightly varied categories yet share a PAS. We assume that an obtained entry is a variation of the canonical one and register the canonical entries in the lexicon. We treat only subject deletion for pro-drop because there is not sufficient information to judge the deletion of other arguments. Scrambling is simply treated as permutation of arguments.

## 4 Evaluation

We used the following for the implementation of our resources: Kyoto corpus ver. 4.0<sup>2</sup>, NAIST text

<sup>2</sup><http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?Kyoto%20University%20Text%20Corpus>

	Training		Develop.		Test	
	St.1	St.2	St.1	St.2	St.1	St.2
Sent.	24,283	24,116	4,833	4,803	9,284	9,245
Converted	24,116	22,820	4,803	4,559	9,245	8,769
Con. rate	99.3	94.6	99.4	94.9	99.6	94.9

Table 5: Statistics of corpus conversion.

Sentential Coverage				
	Covered	Uncovered	Cov. (%)	
Devel.	3,920	639	85.99	
Test	7,610	1,159	86.78	
Lexical Coverage				
	Word	Known	Unknown	
			combi.	cat. word
Devel.	127,144	126,383	682	79
Test	238,083	236,651	1,242	145

Table 6: Sentential and lexical coverage.

corpus ver. 1.5<sup>3</sup>, and JP corpus ver. 1.0<sup>4</sup>. The integrated corpus is divided into training, development, and final test sets following the standard data split in previous works on Japanese dependency parsing (Kudo and Matsumoto, 2002). The details of these resources are shown in Table 4.

### 4.1 Corpus conversion and lexicon extraction

Table 5 shows the number of successful conversions performed by our method. In total, we obtained 22,820 CCG derivations from 24,283 sentences (in the training set), resulting in the total conversion rate of 93.98%. The table shows we lost more sentences in Step 2 than in Step 1. This is natural because Step 2 imposed more restrictions on resulting structures and therefore detected more discrepancies including compounding errors. Our conversion rate is about 5.5 points lower than the English counterpart (Hockenmaier and Steedman, 2007). Manual investigation of the sampled derivations would be beneficial for the conversion improvement.

For the lexicon extraction from the CCGbank, we obtained 699 types of lexical categories from 616,305 word tokens. After lexical reduction, the number of categories decreased to 454, which in turn may produce 5,342 categories by lexical expansion. The average number of categories for a word type was 11.68 as a result.

### 4.2 Evaluation of coverage

Following the evaluation criteria in (Hockenmaier and Steedman, 2007), we measured the coverage

<sup>3</sup><http://cl.naist.jp/nldata/corpus/>

<sup>4</sup><https://alaginrc.nict.go.jp/resources/tocorpus/tocorpusabstract.html>

of the grammar on unseen texts. First, we obtained CCG derivations for evaluation sets by applying our conversion method and then used these derivations as gold standard. Lexical coverage indicates the number of words to which the grammar assigns a gold standard category. Sentential coverage indicates the number of sentences in which all words are assigned gold standard categories <sup>5</sup>.

Table 6 shows the evaluation results. Lexical coverage was 99.40% with rare word treatment, which is in the same level as the case of the English CCG parser C&C (Clark and Curran, 2007). We also measured coverage in a “weak” sense, which means the number of sentences that are given at least one analysis (not necessarily correct) by the obtained grammar. This number was 99.12 % and 99.06 % for the development and the test set, respectively, which is sufficiently high for wide-coverage parsing of real-world texts.

### 4.3 Evaluation of parsing accuracy

Finally, we evaluated the parsing accuracy. We employed the parser and the supertagger of (Miyao and Tsujii, 2008), specifically, its generalized modules for lexicalized grammars. We trained log-linear models in the same way as (Clark and Curran, 2007) using the training set as training data. Feature sets were simply borrowed from an English parser; no tuning was performed. Following conventions in research on Japanese dependency parsing, gold morphological analysis results were input to a parser. Following C&C, the evaluation measure was precision and recall over dependencies, where a dependency is defined as a 4-tuple: a head of a functor, a functor category, an argument slot, and a head of an argument.

Table 7 shows the parsing accuracy on the development and the test sets. The supertagging accuracy is presented in the upper table. While our coverage was almost the same as C&C, the performance of our supertagger and parser was lower. To improve the performance, tuning disambiguation models for Japanese is a possible approach. Comparing the parser’s performance with previous works on Japanese dependency parsing is difficult as our figures are not directly comparable to theirs. Sassano and Kurohashi (2009) reported the accuracy of their parser as 88.48 and 95.09

<sup>5</sup>Since a gold derivation can logically be obtained if gold categories are assigned to all words in a sentence, sentential coverage means that the obtained lexicon has the ability to produce exactly correct derivations for those sentences.

<b>Supertagging accuracy</b>						
	Lex. Cov.		Cat. Acc.			
Devel.	99.40		90.86			
Test	99.40		90.69			
C&C	99.63		94.32			
<b>Overall performance</b>						
	LP	LR	LF	UP	UR	UF
Devel.	82.55	82.73	82.64	90.02	90.22	90.12
Test	82.40	82.59	82.50	89.95	90.15	90.05
C&C	88.34	86.96	87.64	93.74	92.28	93.00

Table 7: Parsing accuracy. LP, LR and LF refer to labeled precision, recall, and F-score respectively. UP, UR, and UF are for unlabeled.

in unlabeled chunk-based and word-based F1 respectively. While our score of 90.05 in unlabeled category dependency seems to be lower than their word-based score, this is reasonable because our category dependency includes more difficult problems, such as whether a subject PP is shared by coordinated verbs. Thus, our parser is expected to be capable of real-world Japanese text analysis as well as dependency parsers.

## 5 Conclusion

In this paper, we proposed a method to induce wide-coverage Japanese resources based on CCG that will lead to deeper syntactic analysis for Japanese and presented empirical evaluation in terms of the quality of the obtained lexicon and the parsing accuracy. Although our work is basically in line with CCGbank, the application of the method to Japanese is not trivial due to the fact that the relationship between chunk-based dependency structures and CCG derivations is not obvious.

Our method integrates multiple dependency-based resources to convert them into an integrated phrase structure treebank. The obtained treebank is then transformed into CCG derivations. The empirical evaluation in Sec. 4 shows that our corpus conversion successfully converts 94 % of the corpus sentences and the coverage of the lexicon is 99.4 %, which is sufficiently high for analyzing real-world texts. A comparison of the parsing accuracy with previous works on Japanese dependency parsing and English CCG parsing indicates that our parser can analyze real-world Japanese texts fairly well and that there is room for improvement in disambiguation models.

## References

- Daisuke Bekki. 2010. *Formal Theory of Japanese Syntax*. Kuroshio Shuppan. (In Japanese).
- Johan Bos, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. 2004. Wide-coverage semantic representations from a CCG parser. In *Proceedings of COLING 2004*, pages 1240–1246.
- Johan Bos, Cristina Bosco, and Alessandro Mazzei. 2009. Converting a dependency treebank to a categorical grammar treebank for Italian. In *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories (TLT8)*, pages 27–38.
- Johan Bos. 2007. Recognising textual entailment and computational semantics. In *Proceedings of Seventh International Workshop on Computational Semantics IWCS-7*, page 1.
- Ruken Çakıcı. 2005. Automatic induction of a CCG grammar for Turkish. In *Proceedings of ACL Student Research Workshop*, pages 73–78.
- Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4).
- Takao Gunji. 1987. *Japanese Phrase Structure Grammar: A Unification-based Approach*. D. Reidel.
- Hiroki Hanaoka, Hideki Mima, and Jun’ichi Tsujii. 2010. A Japanese particle corpus built by example-based annotation. In *Proceedings of LREC 2010*.
- Yuta Hayashibe, Mamoru Komachi, and Yuji Matsumoto. 2011. Japanese predicate argument structure analysis exploiting argument position and type. In *Proceedings of IJCNLP 2011*, pages 201–209.
- Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.
- Julia Hockenmaier. 2006. Creating a CCGbank and a wide-coverage CCG lexicon for German. In *Proceedings of the Joint Conference of COLING/ACL 2006*.
- Ryu Iida and Massimo Poesio. 2011. A cross-lingual ILP solution to zero anaphora resolution. In *Proceedings of ACL-HLT 2011*, pages 804–813.
- Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a Japanese text corpus with predicate-argument and coreference relations. In *Proceedings of Linguistic Annotation Workshop*, pages 132–139.
- Daisuke Kawahara and Sadao Kurohashi. 2011. Generative modeling of coordination by factoring parallelism and selectional preferences. In *Proceedings of IJCNLP 2011*.
- Daisuke Kawahara, Sadao Kurohashi, and Koiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 8th Annual Meeting of the Association for Natural Language Processing*, pages 495–498. (In Japanese).
- Nobo Komagata. 1999. *Information Structure in Texts: A Computational Analysis of Contextual Appropriateness in English and Japanese*. Ph.D. thesis, University of Pennsylvania.
- Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proceedings of CoNLL 2002*.
- M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- Yusuke Miyao and Jun’ichi Tsujii. 2008. Feature forest models for probabilistic HPSG parsing. *Computational Linguistics*, 34(1):35–80.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.
- Ivan A. Sag, Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction, 2nd Edition*. CSLI Publications.
- Ryohei Sasano and Sadao Kurohashi. 2011. A discriminative approach to Japanese zero anaphora resolution with large-scale lexicalized case frames. In *Proceedings of IJCNLP 2011*.
- Manabu Sassano and Sadao Kurohashi. 2009. A unified single scan algorithm for Japanese base phrase chunking and dependency parsing. In *Proceedings of ACL-IJCNLP 2009*.
- Melanie Siegel and Emily M. Bender. 2002. Efficient deep processing of Japanese. In *Proceedings of the 3rd Workshop on Asian Language Resources and International Standardization*.
- Mark Steedman. 2001. *The Syntactic Process*. MIT Press.
- David Vadas and James Curran. 2007. Adding noun phrase structure to the Penn Treebank. In *Proceedings of ACL 2007*, pages 240–247.
- Emiko Yamada, Eiji Aramaki, Takeshi Imai, and Kazuhiko Ohe. 2010. Internal structure of a disease name and its application for ICD coding. *Studies in health technology and informatics*, 160(2):1010–1014.
- Kazuhiro Yoshida. 2005. Corpus-oriented development of Japanese HPSG parsers. In *Proceedings of the ACL Student Research Workshop*.

# Transition-based Dependency Parsing with Selectional Branching

**Jinho D. Choi**

Department of Computer Science  
University of Massachusetts Amherst  
Amherst, MA, 01003, USA  
jdchoi@cs.umass.edu

**Andrew McCallum**

Department of Computer Science  
University of Massachusetts Amherst  
Amherst, MA, 01003, USA  
mccallum@cs.umass.edu

## Abstract

We present a novel approach, called selectional branching, which uses confidence estimates to decide when to employ a beam, providing the accuracy of beam search at speeds close to a greedy transition-based dependency parsing approach. Selectional branching is guaranteed to perform a fewer number of transitions than beam search yet performs as accurately. We also present a new transition-based dependency parsing algorithm that gives a complexity of  $O(n)$  for projective parsing and an expected linear time speed for non-projective parsing. With the standard setup, our parser shows an unlabeled attachment score of 92.96% and a parsing speed of 9 milliseconds per sentence, which is faster and more accurate than the current state-of-the-art transition-based parser that uses beam search.

## 1 Introduction

Transition-based dependency parsing has gained considerable interest because it runs fast and performs accurately. Transition-based parsing gives complexities as low as  $O(n)$  and  $O(n^2)$  for projective and non-projective parsing, respectively (Nivre, 2008).<sup>1</sup> The complexity is lower for projective parsing because a parser can deterministically skip tokens violating projectivity, while this property is not assumed for non-projective parsing. Nonetheless, it is possible to perform non-projective parsing in expected linear time because the amount of non-projective dependencies is notably smaller (Nivre and Nilsson, 2005) so a parser can assume projectivity for most cases while recognizing ones for which projectivity should not be assumed (Nivre, 2009; Choi and Palmer, 2011).

<sup>1</sup>We refer parsing approaches that produce only projective dependency trees as projective parsing and both projective and non-projective dependency trees as non-projective parsing.

Greedy transition-based dependency parsing has been widely deployed because of its speed (Cer et al., 2010); however, state-of-the-art accuracies have been achieved by globally optimized parsers using beam search (Zhang and Clark, 2008; Huang and Sagae, 2010; Zhang and Nivre, 2011; Bohnet and Nivre, 2012). These approaches generate multiple transition sequences given a sentence, and pick one with the highest confidence. Coupled with dynamic programming, transition-based dependency parsing with beam search can be done very efficiently and gives significant improvement to parsing accuracy.

One downside of beam search is that it always uses a fixed size of beam even when a smaller size of beam is sufficient for good results. In our experiments, a greedy parser performs as accurately as a parser that uses beam search for about 64% of time. Thus, it is preferred if the beam size is not fixed but proportional to the number of low confidence predictions that a greedy parser makes, in which case, fewer transition sequences need to be explored to produce the same or similar parse output.

We first present a new transition-based parsing algorithm that gives a complexity of  $O(n)$  for projective parsing and an expected linear time speed for non-projective parsing. We then introduce selectional branching that uses confidence estimates to decide when to employ a beam. With our new approach, we achieve a higher parsing accuracy than the current state-of-the-art transition-based parser that uses beam search and a much faster speed.

## 2 Transition-based dependency parsing

We introduce a transition-based dependency parsing algorithm that is a hybrid between Nivre’s arc-eager and list-based algorithms (Nivre, 2003; Nivre, 2008). Nivre’s arc-eager is a projective parsing algorithm showing a complexity of  $O(n)$ . Nivre’s list-based algorithm is a non-projective parsing algorithm showing a complexity of  $O(n^2)$ . Table 1 shows transitions in our algorithm. The top 4 and

Transition	Current state	⇒	Resulting state
LEFT <sub>l</sub> -REDUCE	$([\sigma i], \delta, [j \beta], A)$	⇒	$(\sigma, \delta, [j \beta], A \cup \{i \xleftarrow{l} j\})$
RIGHT <sub>l</sub> -SHIFT	$([\sigma i], \delta, [j \beta], A)$	⇒	$([\sigma i\delta]j, [], \beta, A \cup \{i \xrightarrow{l} j\})$
NO-SHIFT	$([\sigma i], \delta, [j \beta], A)$	⇒	$([\sigma i\delta]j, [], \beta, A)$
NO-REDUCE	$([\sigma i], \delta, [j \beta], A)$	⇒	$(\sigma, \delta, [j \beta], A)$
LEFT <sub>l</sub> -PASS	$([\sigma i], \delta, [j \beta], A)$	⇒	$(\sigma, [i \delta], [j \beta], A \cup \{i \xleftarrow{l} j\})$
RIGHT <sub>l</sub> -PASS	$([\sigma i], \delta, [j \beta], A)$	⇒	$(\sigma, [i \delta], [j \beta], A \cup \{i \xrightarrow{l} j\})$
NO-PASS	$([\sigma i], \delta, [j \beta], A)$	⇒	$(\sigma, [i \delta], [j \beta], A)$

Table 1: Transitions in our dependency parsing algorithm.

Transition	Preconditions
LEFT <sub>l</sub> -*	$[i \neq 0] \wedge \neg[\exists k. (i \leftarrow k) \in A] \wedge \neg[(i \rightarrow^* j) \in A]$
RIGHT <sub>l</sub> -*	$\neg[\exists k. (k \rightarrow j) \in A] \wedge \neg[(i \leftarrow^* j) \in A]$
*-SHIFT	$\neg[\exists k \in \sigma. (k \neq i) \wedge ((k \leftarrow j) \vee (k \rightarrow j))]$
*-REDUCE	$[\exists h. (h \rightarrow i) \in A] \wedge \neg[\exists k \in \beta. (i \rightarrow k)]$

Table 2: Preconditions of the transitions in Table 1 (\* is a wildcard representing any transition).

the bottom 3 transitions are inherited from Nivre’s arc-eager and list-based algorithms, respectively.<sup>2</sup>

Each parsing state is represented as a tuple  $(\sigma, \delta, \beta, A)$ , where  $\sigma$  is a stack containing processed tokens,  $\delta$  is a deque containing tokens popped out of  $\sigma$  but will be pushed back into  $\sigma$  in later parsing states to handle non-projectivity, and  $\beta$  is a buffer containing unprocessed tokens.  $A$  is a set of labeled arcs.  $(i, j)$  represent indices of their corresponding tokens ( $w_i, w_j$ ),  $l$  is a dependency label, and the 0 identifier corresponds to  $w_0$ , introduced as the root of a tree. The initial state is  $([0], [], [1, \dots, n], \emptyset)$ , and the final state is  $(\sigma, \delta, [], A)$ . At any parsing state, a decision is made by comparing the top of  $\sigma$ ,  $w_i$ , and the first element of  $\beta$ ,  $w_j$ . This decision is consulted by gold-standard trees during training and a classifier during decoding.

LEFT<sub>l</sub>-\* and RIGHT<sub>l</sub>-\* are performed when  $w_j$  is the head of  $w_i$  with a dependency label  $l$ , and vice versa. After LEFT<sub>l</sub>-\* or RIGHT<sub>l</sub>-, an arc is added to  $A$ . NO-\* is performed when no dependency is found for  $w_i$  and  $w_j$ . \*-SHIFT is performed when no dependency is found for  $w_j$  and any token in  $\sigma$  other than  $w_i$ . After \*-SHIFT, all tokens in  $\delta$  as well as  $w_j$  are pushed into  $\sigma$ . \*-REDUCE is performed when  $w_i$  already has the head, and  $w_i$  is not the head of any token in  $\beta$ . After \*-REDUCE,  $w_i$  is popped out of  $\sigma$ . \*-PASS is performed when neither \*-SHIFT nor \*-REDUCE can be performed. After \*-PASS,  $w_i$  is moved to the front of  $\delta$  so it

can be compared to other tokens in  $\beta$  later. Each transition needs to satisfy certain preconditions to ensure the properties of a well-formed dependency graph (Nivre, 2008); they are described in Table 2.  $(i \leftarrow j)$  and  $(i \leftarrow^* j)$  indicate that  $w_j$  is the head and an ancestor of  $w_i$  with any label, respectively.

When a parser is trained on only projective trees, our algorithm learns only the top 4 transitions and produces only projective trees during decoding. In this case, it performs at most  $2n - 1$  transitions per sentence so the complexity is  $O(n)$ . When a parser is trained on a mixture of projective and non-projective trees, our algorithm learns all transitions and produces both kinds of trees during decoding. In this case, it performs at most  $\frac{n(n+1)}{2}$  transitions so the complexity is  $O(n^2)$ . However, because of the presence of \*-SHIFT and \*-REDUCE, our algorithm is capable of skipping or removing tokens during non-projective parsing, which allows it to show a linear time parsing speed in practice.

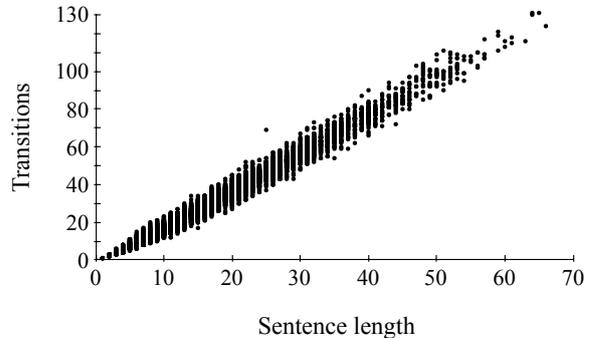
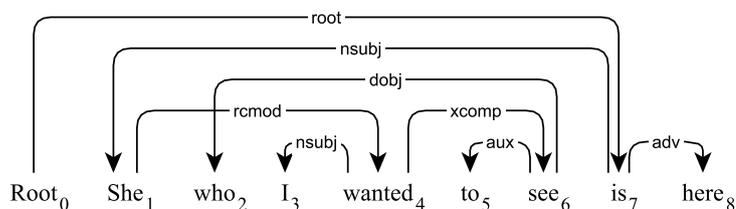


Figure 1: The # of transitions performed during training with respect to sentence lengths for Dutch.

<sup>2</sup>The parsing complexity of a transition-based dependency parsing algorithm is determined by the number of transitions performed with respect to the number of tokens in a sentence, say  $n$  (Kübler et al., 2009).



	Transition	$\sigma$	$\delta$	$\beta$	$A$
0	Initialization	[0]	[]	[1  $\beta$ ]	$\emptyset$
1	NO-SHIFT	$[\sigma 1]$	[]	[2  $\beta$ ]	
2	NO-SHIFT	$[\sigma 2]$	[]	[3  $\beta$ ]	
3	NO-SHIFT	$[\sigma 3]$	[]	[4  $\beta$ ]	
4	LEFT-REDUCE	$[\sigma 2]$	[]	[4  $\beta$ ]	$A \cup \{3 \leftarrow \text{NSUBJ} - 4\}$
5	NO-PASS	$[\sigma 1]$	[2]	[4  $\beta$ ]	
6	RIGHT-SHIFT	$[\sigma 4]$	[]	[5  $\beta$ ]	$A \cup \{1 - \text{RCMOD} \rightarrow 4\}$
7	NO-SHIFT	$[\sigma 5]$	[]	[6  $\beta$ ]	
8	LEFT-REDUCE	$[\sigma 4]$	[]	[6  $\beta$ ]	$A \cup \{5 \leftarrow \text{AUX} - 6\}$
9	RIGHT-PASS	$[\sigma 2]$	[4]	[6  $\beta$ ]	$A \cup \{4 - \text{XCOMP} \rightarrow 6\}$
10	LEFT-REDUCE	$[\sigma 1]$	[4]	[6  $\beta$ ]	$A \cup \{2 \leftarrow \text{DOBJ} - 6\}$
11	NO-SHIFT	$[\sigma 6]$	[]	[7  $\beta$ ]	
12	NO-REDUCE	$[\sigma 4]$	[]	[7  $\beta$ ]	
13	NO-REDUCE	$[\sigma 1]$	[]	[7  $\beta$ ]	
14	LEFT-REDUCE	[0]	[]	[7  $\beta$ ]	$A \cup \{1 \leftarrow \text{NSUBJ} - 7\}$
15	RIGHT-SHIFT	$[\sigma 7]$	[]	[8]	$A \cup \{0 - \text{ROOT} \rightarrow 7\}$
16	RIGHT-SHIFT	$[\sigma 8]$	[]	[]	$A \cup \{7 - \text{ADV} \rightarrow 8\}$

Table 3: A transition sequence generated by our parsing algorithm using gold-standard decisions.

Figure 1 shows the total number of transitions performed during training with respect to sentence lengths for Dutch. Among all languages distributed by the CoNLL-X shared task (Buchholz and Marsi, 2006), Dutch consists of the highest number of non-projective dependencies (5.4% in arcs, 36.4% in trees). Even with such a high number of non-projective dependencies, our parsing algorithm still shows a linear growth in transitions.

Table 3 shows a transition sequence generated by our parsing algorithm using gold-standard decisions. After  $w_3$  and  $w_4$  are compared,  $w_3$  is popped out of  $\sigma$  (state 4) so it is not compared to any other token in  $\beta$  (states 9 and 13). After  $w_2$  and  $w_4$  are compared,  $w_2$  is moved to  $\delta$  (state 5) so it can be compared to other tokens in  $\beta$  (state 10). After  $w_4$  and  $w_6$  are compared, RIGHT-PASS is performed (state 9) because there is a dependency between  $w_6$  and  $w_2$  in  $\sigma$  (state 10). After  $w_6$  and  $w_7$  are compared,  $w_6$  is popped out of  $\sigma$  (state 12) because it is not needed for later parsing states.

### 3 Selectional branching

#### 3.1 Motivation

For transition-based parsing, state-of-the-art accuracies have been achieved by parsers optimized on multiple transition sequences using beam search,

which can be done very efficiently when it is coupled with dynamic programming (Zhang and Clark, 2008; Huang and Sagae, 2010; Zhang and Nivre, 2011; Huang et al., 2012; Bohnet and Nivre, 2012). Despite all the benefits, there is one downside of this approach; it generates a fixed number of transition sequences no matter how confident the one-best sequence is.<sup>3</sup> If every prediction leading to the one-best sequence is confident, it may not be necessary to explore more sequences to get the best output. Thus, it is preferred if the beam size is not fixed but proportional to the number of low confidence predictions made for the one-best sequence.

The selectional branching method presented here performs at most  $d \cdot t - e$  transitions, where  $t$  is the maximum number of transitions performed to generate a transition sequence,  $d = \min(b, |\lambda| + 1)$ ,  $b$  is the beam size,  $|\lambda|$  is the number of low confidence predictions made for the one-best sequence, and  $e = \frac{d(d-1)}{2}$ . Compared to beam search that always performs  $b \cdot t$  transitions, selectional branching is guaranteed to perform fewer transitions given the same beam size because  $d \leq b$  and  $e > 0$  except for  $d = 1$ , in which case, no branching happens. With selectional branching, our parser shows slightly

<sup>3</sup>The ‘one-best sequence’ is a transition sequence generated by taking only the best prediction at each parsing state.

higher parsing accuracy than the current state-of-the-art transition-based parser using beam search, and performs about 3 times faster.

### 3.2 Branching strategy

Figure 2 shows an overview of our branching strategy.  $s_{ij}$  represents a parsing state, where  $i$  is the index of the current transition sequence and  $j$  is the index of the current parsing state (e.g.,  $s_{12}$  represents the 2nd parsing state in the 1st transition sequence).  $p_{kj}$  represents the  $k$ 'th best prediction (in our case, it is a predicted transition) given  $s_{1j}$  (e.g.,  $p_{21}$  is the 2nd-best prediction given  $s_{11}$ ).

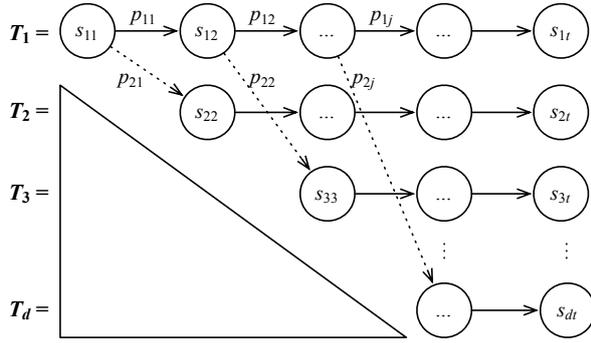


Figure 2: An overview of our branching strategy. Each sequence  $T_{i>1}$  branches from  $T_1$ .

Initially, the one-best sequence  $T_1 = [s_{11}, \dots, s_{1t}]$  is generated by a greedy parser. While generating  $T_1$ , the parser adds tuples  $(s_{1j}, p_{2j}), \dots, (s_{1j}, p_{kj})$  to a list  $\lambda$  for each low confidence prediction  $p_{1j}$  given  $s_{1j}$ .<sup>4</sup> Then, new transition sequences are generated by using the  $b$  highest scoring predictions in  $\lambda$ , where  $b$  is the beam size. If  $|\lambda| < b$ , all predictions in  $\lambda$  are used. The same greedy parser is used to generate these new sequences although it now starts with  $s_{1j}$  instead of an initial parsing state, applies  $p_{kj}$  to  $s_{1j}$ , and performs further transitions. Once all transition sequences are generated, a parse tree is built from a sequence with the highest score.

For our experiments, we set  $k = 2$ , which gave noticeably more accurate results than  $k = 1$ . We also experimented with  $k > 2$ , which did not show significant improvement over  $k = 2$ . Note that assigning a greater  $k$  may increase  $|\lambda|$  but not the total number of transition sequences generated, which is restricted by the beam size,  $b$ . Since each sequence  $T_{i>1}$  branches from  $T_1$ , selectional branching performs fewer transitions than beam search: at least  $\frac{d(d-1)}{2}$  transitions are inherited from  $T_1$ ,

<sup>4</sup> $\lambda$  is initially empty, which is hidden in Figure 2.

where  $d = \min(b, |\lambda| + 1)$ ; thus, it performs that many transitions less than beam search (see the left lower triangle in Figure 2). Furthermore, selectional branching generates a  $d$  number of sequences, where  $d$  is proportional to the number of low confidence predictions made by  $T_1$ . To sum up, selectional branching generates the same or fewer transition sequences than beam search and each sequence  $T_{i>1}$  performs fewer transitions than  $T_1$ ; thus, it performs faster than beam search in general given the same beam size.

### 3.3 Finding low confidence predictions

For each parsing state  $s_{ij}$ , a prediction is made by generating a feature vector  $x_{ij} \in \mathcal{X}$ , feeding it into a classifier  $C^1$  that uses a feature map  $\Phi(x, y)$  and a weight vector  $\mathbf{w}$  to measure a score for each label  $y \in \mathcal{Y}$ , and choosing a label with the highest score. When there is a tie between labels with the highest score, the first one is chosen. This can be expressed as a logistic regression:

$$C^1(x) = \arg \max_{y \in \mathcal{Y}} \{f(x, y)\}$$

$$f(x, y) = \frac{\exp(\mathbf{w} \cdot \Phi(x, y))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w} \cdot \Phi(x, y'))}$$

To find low confidence predictions, we use the margins (score differences) between the best prediction and the other predictions. If all margins are greater than a threshold, the best prediction is considered highly confident; otherwise, it is not. Given this analogy, the  $k$ -best predictions can be found as follows ( $m \geq 0$  is a margin threshold):

$$C^k(x, m) = \mathcal{K} \arg \max_{y \in \mathcal{Y}} \{f(x, y)\}$$

$$\text{s.t. } f(x, C^1(x)) - f(x, y) \leq m$$

' $\mathcal{K} \arg \max$ ' returns a set of  $k'$  labels whose margins to  $C^1(x)$  are smaller than any other label's margin to  $C^1(x)$  and also  $\leq m$ , where  $k' \leq k$ . When  $m = 0$ , it returns a set of the highest scoring labels only, including  $C^1(x)$ . When  $m = 1$ , it returns a set of all labels. Given this, a prediction is considered not confident if  $|C^k(x, m)| > 1$ .

### 3.4 Finding the best transition sequence

Let  $P_i$  be a list of all predictions that lead to generate a transition sequence  $T_i$ . The predictions in  $P_i$  are either inherited from  $T_1$  or made specifically for  $T_i$ . In Figure 2,  $P_3$  consists of  $p_{11}$  as its first prediction,  $p_{22}$  as its second prediction, and

further predictions made specifically for  $T_3$ . The score of each prediction is measured by  $f(x, y)$  in Section 3.3. Then, the score of  $T_i$  is measured by averaging scores of all predictions in  $P_i$ .

$$\text{score}(T_i) = \frac{\sum_{p \in P_i} \text{score}(p)}{|P_i|}$$

Unlike Zhang and Clark (2008), we take the average instead of the sum of all prediction scores. This is because our algorithm does not guarantee the same number of transitions for every sequence, so the sum of all scores would weigh more on sequences with more transitions. We experimented with both the sum and the average, and taking the average led to slightly higher parsing accuracy.

### 3.5 Bootstrapping transition sequences

During training, a training instance is generated for each parsing state  $s_{ij}$  by taking a feature vector  $x_{ij}$  and its true label  $y_{ij}$ . To generate multiple transition sequences during training, the bootstrapping technique of Choi and Palmer (2011) is used, which is described in Algorithm 1.<sup>5</sup>

---

#### Algorithm 1 Bootstrapping

---

**Input:**  $D_t$ : training set,  $D_d$ : development set.  
**Output:** A model  $M$ .

- 1:  $r \leftarrow 0$
- 2:  $I \leftarrow \text{getTrainingInstances}(D_t)$
- 3:  $M_0 \leftarrow \text{buildModel}(I)$
- 4:  $S_0 \leftarrow \text{getScore}(D_d, M_0)$
- 5: **while** ( $r = 0$ ) or ( $S_{r-1} < S_r$ ) **do**
- 6:    $r \leftarrow r + 1$
- 7:    $I \leftarrow \text{getTrainingInstances}(D_t, M_{r-1})$
- 8:    $M_r \leftarrow \text{buildModel}(I)$
- 9:    $S_r \leftarrow \text{getScore}(D_d, M_r)$
- 10: **return**  $M_{r-1}$

---

First, an initial model  $M_0$  is trained on all data by taking the one-best sequences, and its score is measured by testing on a development set (lines 2-4). Then, the next model  $M_r$  is trained on all data but this time,  $M_{r-1}$  is used to generate multiple transition sequences (line 7-8). Among all transition sequences generated by  $M_{r-1}$ , training instances from only  $T_1$  and  $T_g$  are used to train  $M_r$ , where  $T_1$  is the one-best sequence and  $T_g$  is a sequence giving the most accurate parse output compared to the gold-standard tree. The score of  $M_r$  is measured (line 9), and repeat the procedure if  $S_{r-1} < S_r$ ; otherwise, return the previous model  $M_{r-1}$ .

<sup>5</sup>Alternatively, the dynamic oracle approach of Goldberg and Nivre (2012) can be used to generate multiple transition sequences, which is expected to show similar results.

### 3.6 Adaptive subgradient algorithm

To build each model during bootstrapping, we use a stochastic adaptive subgradient algorithm called ADAGRAD that uses per-coordinate learning rates to exploit rarely seen features while remaining scalable (Duchi et al., 2011). This is suitable for NLP tasks where rarely seen features often play an important role and training data consists of a large number of instances with high dimensional features. Algorithm 2 shows our adaptation of ADAGRAD with logistic regression for multi-class classification. Note that when used with logistic regression, ADAGRAD takes a regular gradient instead of a subgradient method for updating weights. For our experiments, ADAGRAD slightly outperformed learning algorithms such as average perceptron (Collins, 2002) or Liblinear SVM (Hsieh et al., 2008).

---

#### Algorithm 2 ADAGRAD + logistic regression

---

**Input:**  $D = \{(x_i, y_i)\}_{i=1}^n$  s.t.  $x_i \in \mathcal{X}, y_i \in \mathcal{Y}$   
 $\Phi(x, y) \in \mathbb{R}^d$  s.t.  $d = \text{dimension}(\mathcal{X}) \times |\mathcal{Y}|$   
 $T$ : iterations,  $\alpha$ : learning rate,  $\rho$ : ridge  
**Output:** A weight vector  $\mathbf{w} \in \mathbb{R}^d$ .

- 1:  $\mathbf{w} \leftarrow 0$ , where  $\mathbf{w} \in \mathbb{R}^d$
- 2:  $\mathbf{G} \leftarrow 0$ , where  $\mathbf{G} \in \mathbb{R}^d$
- 3: **for**  $t \leftarrow 1 \dots T$  **do**
- 4:   **for**  $i \leftarrow 1 \dots n$  **do**
- 5:      $\mathbf{Q}_{y \in \mathcal{Y}} \leftarrow I(y_i, y) - f(x_i, y)$ , s.t.  $\mathbf{Q} \in \mathbb{R}^{|\mathcal{Y}|}$
- 6:      $\partial \leftarrow \sum_{y \in \mathcal{Y}} (\Phi(x_i, y) \cdot \mathbf{Q}_y)$
- 7:      $\mathbf{G} \leftarrow \mathbf{G} + \partial \circ \partial$
- 8:     **for**  $j \leftarrow 1 \dots d$  **do**
- 9:        $\mathbf{w}_j \leftarrow \mathbf{w}_j + \alpha \cdot \frac{1}{\rho + \sqrt{\mathbf{G}_j}} \cdot \partial_j$

---

$$I(y, y') = \begin{cases} 1 & y = y' \\ 0 & \text{otherwise} \end{cases}$$

The algorithm takes three hyper-parameters;  $T$  is the number of iterations,  $\alpha$  is the learning rate, and  $\rho$  is the ridge ( $T > 0, \alpha > 0, \rho \geq 0$ ).  $\mathbf{G}$  is our running estimate of a diagonal covariance matrix for the gradients (per-coordinate learning rates). For each instance, scores for all labels are measured by the logistic regression function  $f(x, y)$  in Section 3.3. These scores are subtracted from an output of the indicator function  $I(y, y')$ , which forces our model to keep learning this instance until the prediction is 100% confident (in other words, until the score of  $y_i$  becomes 1). Then, a subgradient is measured by taking all feature vectors together weighted by  $\mathbf{Q}$  (line 6). This subgradient is used to update  $\mathbf{G}$  and  $\mathbf{w}$ , where  $\circ$  is the Hadamard product (lines 7-9).  $\rho$  is a ridge term to keep the inverse covariance well-conditioned.

## 4 Experiments

### 4.1 Corpora

For projective parsing experiments, the Penn English Treebank (Marcus et al., 1993) is used with the standard split: sections 2-21 for training, 22 for development, and 23 for evaluation. All constituent trees are converted with the head-finding rules of Yamada and Matsumoto (2003) and the labeling rules of Nivre (2006). For non-projective parsing experiments, four languages from the CoNLL-X shared task are used: Danish, Dutch, Slovene, and Swedish (Buchholz and Marsi, 2006). These languages are selected because they contain non-projective trees and are publicly available from the CoNLL-X webpage.<sup>6</sup> Since the CoNLL-X data we have does not come with development sets, the last 10% of each training set is used for development.

### 4.2 Feature engineering

For English, we mostly adapt features from Zhang and Nivre (2011) who have shown state-of-the-art parsing accuracy for transition-based dependency parsing. Their distance features are not included in our approach because they do not seem to show meaningful improvement. Feature selection is done on the English development set.

For the other languages, the same features are used with the addition of morphological features provided by CoNLL-X; specifically, morphological features from the top of  $\sigma$  and the front of  $\beta$  are added as unigram features. Moreover, all POS tag features from English are duplicated with coarse-grained POS tags provided by CoNLL-X. No more feature engineering is done for these languages; it is possible to achieve higher performance by using different features, especially when these languages contain non-projective dependencies whereas English does not, which we will explore in the future.

### 4.3 Development

Several parameters need to be optimized during development. For ADAGRAD,  $T$ ,  $\alpha$ , and  $\rho$  need to be tuned (Section 3.6). For bootstrapping, the number of iterations, say  $r$ , needs to be tuned (Section 3.5). For selectional branching, the margin threshold  $m$  and the beam size  $b$  need to be tuned (Section 3.3). First, all parameters are tuned on the English development set by using grid search on  $T = [1, \dots, 10]$ ,  $\alpha = [0, 01, 0, 02]$ ,  $\rho = [0.1, 0.2]$ ,  $r = [1, 2, 3]$ ,

$m = [0.83, \dots, 0.92]$ , and  $b = [16, 32, 64, 80]$ . As a result, the following parameters are found:  $\alpha = 0.02$ ,  $\rho = 0.1$ ,  $m = 0.88$ , and  $b = 64|80$ . For this development set, the beam size of 64 and 80 gave the exact same result, so we kept the one with a larger beam size ( $b = 80$ ).

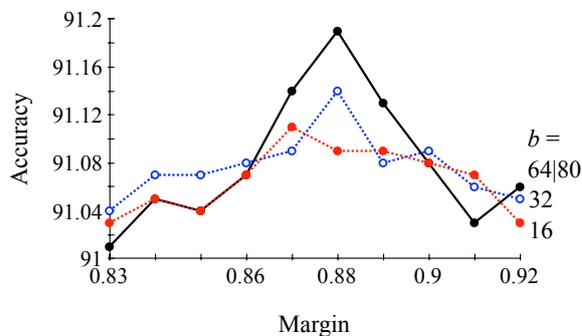


Figure 3: Parsing accuracies with respect to margins and beam sizes on the English development set.  $b = 64|80$ : the black solid line with solid circles,  $b = 32$ : the blue dotted line with hollow circles,  $b = 16$ : the red dotted line with solid circles.

Figure 3 shows parsing accuracies with respect to different margins and beam sizes on the English development set. These parameters need to be tuned jointly because different margins prefer different beam sizes. For instance,  $m = 0.85$  gives the highest accuracy with  $b = 32$ , but  $m = 0.88$  gives the highest accuracy with  $b = 64|80$ .

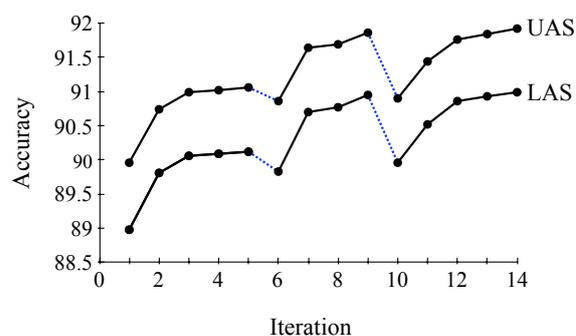


Figure 4: Parsing accuracies with respect to ADAGRAD and bootstrap iterations on the English development set when  $\alpha = 0.02$ ,  $\rho = 0.1$ ,  $m = 0.88$ , and  $b = 64|80$ . UAS: unlabeled attachment score, LAS: labeled attachment score.

Figure 4 shows parsing accuracies with respect to ADAGRAD and bootstrap iterations on the English development set. The range 1-5 shows results of 5 ADAGRAD iterations before bootstrapping, the range 6-9 shows results of 4 iterations during the

<sup>6</sup><http://ilk.uvt.nl/conll/>

first bootstrapping, and the range 10-14 shows results of 5 iterations during the second bootstrapping. Thus, the number of bootstrap iteration is 2 where each bootstrapping takes a different number of ADAGRAD iterations. Using an Intel Xeon 2.57GHz machine, it takes less than 40 minutes to train the entire Penn Treebank, which includes times for IO, feature extraction and bootstrapping.

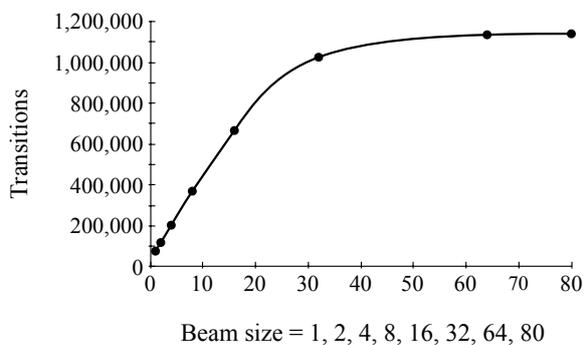


Figure 5: The total number of transitions performed during decoding with respect to beam sizes on the English development set.

Figure 5 shows the total number of transitions performed during decoding with respect to beam sizes on the English development set (1,700 sentences, 40,117 tokens). With selectional branching, the number of transitions grows logarithmically as the beam size increases whereas it would have grown linearly if beam search were used. We also checked how often the one best sequence is chosen as the final sequence during decoding. Out of 1,700 sentences, the one best sequences are chosen for 1,095 sentences. This implies that about 64% of time, our greedy parser performs as accurately as our non-greedy parser using selectional branching.

For the other languages, we use the same values as English for  $\alpha$ ,  $\rho$ ,  $m$ , and  $b$ ; only the ADAGRAD and bootstrap iterations are tuned on the development sets of the other languages.

#### 4.4 Projective parsing experiments

Before parsing, POS tags were assigned to the training set by using 20-way jackknifing. For the automatic generation of POS tags, we used the domain-specific model of Choi and Palmer (2012a)’s tagger, which gave 97.5% accuracy on the English evaluation set (0.2% higher than Collins (2002)’s tagger).

Table 4 shows comparison between past and current state-of-the-art parsers and our approach. The first block shows results from transition-based de-

pendency parsers using beam search. The second block shows results from other kinds of parsing approaches (e.g., graph-based parsing, ensemble parsing, linear programming, dual decomposition). The third block shows results from parsers using external data. The last block shows results from our approach. The Time column show how many seconds per sentence each parser takes.<sup>7</sup>

Approach	UAS	LAS	Time
Zhang and Clark (2008)	92.1		
Huang and Sagae (2010)	92.1		0.04
Zhang and Nivre (2011)	92.9	91.8	0.03
Bohnet and Nivre (2012)	93.38	92.44	0.4
McDonald et al. (2005)	90.9		
McDonald and Pereira (2006)	91.5		
Sagae and Lavie (2006)	92.7		
Koo and Collins (2010)	93.04		
Zhang and McDonald (2012)	93.06	91.86	
Martins et al. (2010)	93.26		
Rush et al. (2010)	93.8		
Koo et al. (2008)	93.16		
Carreras et al. (2008)	93.54		
Bohnet and Nivre (2012)	93.67	92.68	
Suzuki et al. (2009)	93.79		
$b_t = 80, b_d = 80, m = 0.88$	<b>92.96</b>	<b>91.93</b>	<b>0.009</b>
$b_t = 80, b_d = 64, m = 0.88$	92.96	91.93	0.009
$b_t = 80, b_d = 32, m = 0.88$	92.96	91.94	0.009
$b_t = 80, b_d = 16, m = 0.88$	92.96	91.94	0.008
$b_t = 80, b_d = 8, m = 0.88$	92.89	91.87	0.006
$b_t = 80, b_d = 4, m = 0.88$	92.76	91.76	0.004
$b_t = 80, b_d = 2, m = 0.88$	92.56	91.54	0.003
$b_t = 80, b_d = 1, m = 0.88$	92.26	91.25	0.002
$b_t = 1, b_d = 1, m = 0.88$	92.06	91.05	0.002

Table 4: Parsing accuracies and speeds on the English evaluation set, excluding tokens containing only punctuation.  $b_t$  and  $b_d$  indicate the beam sizes used during training and decoding, respectively. UAS: unlabeled attachment score, LAS: labeled attachment score, Time: seconds per sentence.

For evaluation, we use the model trained with  $b = 80$  and  $m = 0.88$ , which is the best setting found during development. Our parser shows higher accuracy than Zhang and Nivre (2011), which is the current state-of-the-art transition-based parser that uses beam search. Bohnet and Nivre (2012)’s transition-based system jointly performs POS tagging and dependency parsing, which shows higher accuracy than ours. Our parser gives a comparative accuracy to Koo and Collins (2010) that is a 3rd-order graph-based parsing approach. In terms of speed, our parser outperforms all other transition-based parsers; it takes about 9 milliseconds per

<sup>7</sup>Dhillon et al. (2012) and Rush and Petrov (2012) also have shown good results on this data but they are excluded from our comparison because they use different kinds of constituent-to-dependency conversion methods.

Approach	Danish		Dutch		Slovene		Swedish	
	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS
Nivre et al. (2006)	84.77	89.80	78.59	81.35	70.30	78.72	84.58	89.50
McDonald et al. (2006)	84.79	90.58	79.19	83.57	73.44	83.17	82.55	88.93
Nivre (2009)	84.2	-	-	-	75.2	-	-	-
F.-González and G.-Rodríguez (2012)	85.17	90.10	-	-	-	-	83.55	89.30
Nivre and McDonald (2008)	86.67	-	81.63	-	-	75.94	84.66	-
Martins et al. (2010)	-	<b>91.50</b>	-	84.91	-	<b>85.53</b>	-	89.80
$b_t = 80, b_d = 1, m = 0.88$	86.75	91.04	80.75	83.59	75.66	83.29	86.32	91.12
$b_t = 80, b_d = 80, m = 0.88$	<b>87.27</b>	91.36	<b>82.45</b>	<b>85.33</b>	<b>77.46</b>	84.65	<b>86.80</b>	<b>91.36</b>

Table 5: Parsing accuracies on four languages with non-projective dependencies, excluding punctuation.

sentence using the beam size of 80. Our parser is implemented in Java and tested on an Intel Xeon 2.57GHz. Note that we do not include input/output time for our speed comparison.

For a proof of concept, we run the same model, trained with  $b_t = 80$ , but decode with different beam sizes using the same margin. Surprisingly, our parser gives the same accuracy (0.01% higher for labeled attachment score) on this data even with  $b_d = 16$ . More importantly,  $b_d = 16$  shows about the same parsing speed as  $b_d = 80$ , which indicates that selectional branching automatically reduced down the beam size by estimating low confidence predictions, so even if we assigned a larger beam size for decoding, it would have performed as efficiently. This implies that we no longer need to be so conscious about the beam size during decoding.

Another interesting part is that ( $b_t = 80, b_d = 1$ ) shows higher accuracy than ( $b_t = 1, b_d = 1$ ); this implies that our training method of bootstrapping transition sequences can improve even a greedy parser. Notice that our greedy parser shows higher accuracy than many other greedy parsers (Hall et al., 2006; Goldberg and Elhadad, 2010) because it uses the non-local features of Zhang and Nivre (2011) and the bootstrapping technique of Choi and Palmer (2011) that had not been used for most other greedy parsing approaches.

#### 4.5 Non-projective parsing experiments

Table 5 shows comparison between state-of-the-art parsers and our approach for four languages with non-projective dependencies. Nivre et al. (2006) uses a pseudo-projective transition-based parsing approach. McDonald et al. (2006) uses a 2nd-order maximum spanning tree approach. Nivre (2009) and Fernández-González and Gómez-Rodríguez (2012) use different non-projective transition-based parsing approaches. Nivre and McDonald (2008) uses an ensemble model between transition-based and graph-based parsing approaches. Martins et

al. (2010) uses integer linear programming for the optimization of their parsing model.

Some of these approaches use greedy parsers, so we include our results from models using ( $b_t = 80, b_d = 1, m = 0.88$ ), which finds only the one-best sequences during decoding although it is trained on multiple transition sequences (see Section 4.4). Our parser shows higher accuracies for most languages except for unlabeled attachment scores in Danish and Slovene. Our greedy approach outperforms both Nivre (2009) and Fernández-González and Gómez-Rodríguez (2012) who use different non-projective parsing algorithms.

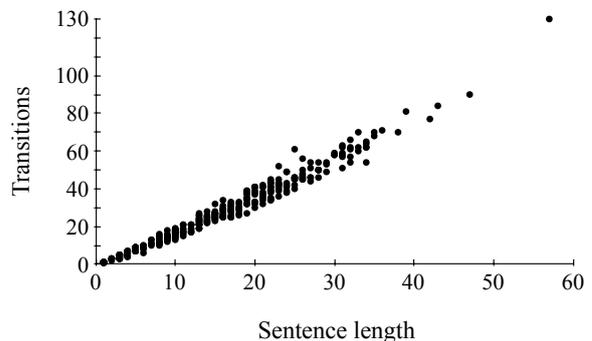


Figure 6: The # of transitions performed during decoding with respect to sentence lengths for Dutch.

Figure 6 shows the number of transitions performed during decoding with respect to sentence lengths for Dutch using  $b_d = 1$ . Our parser still shows a linear growth in transition during decoding.

## 5 Related work

Our parsing algorithm is most similar to Choi and Palmer (2011) who integrated our LEFT-REDUCE transition into Nivre’s list-based algorithm. Our algorithm is distinguished from theirs because ours gives different parsing complexities of  $O(n)$  and  $O(n^2)$  for projective and non-projective parsing, respectively, whereas their algorithm gives  $O(n^2)$

for both cases; this is possible because of the new integration of the RIGHT-SHIFT and NO-REDUCE transitions. There are other transition-based dependency parsing algorithms that take a similar approach; Nivre (2009) integrated a SWAP transition into Nivre’s arc-standard algorithm (Nivre, 2004) and Fernández-González and Gómez-Rodríguez (2012) integrated a buffer transition into Nivre’s arc-eager algorithm to handle non-projectivity.

Our selectional branching method is most relevant to Zhang and Clark (2008) who introduced a transition-based dependency parsing model that uses beam search. Huang and Sagae (2010) later applied dynamic programming to this approach and showed improved efficiency. Zhang and Nivre (2011) added non-local features to this approach and showed improved parsing accuracy. Bohnet and Nivre (2012) introduced a transition-based system that jointly performed POS tagging and dependency parsing. Our work is distinguished from theirs because we use selectional branching instead.

## 6 Conclusion

We present selectional branching that uses confidence estimates to decide when to employ a beam. Coupled with our new hybrid parsing algorithm, ADAGRAD, rich non-local features, and bootstrapping, our parser gives higher parsing accuracy than most other transition-based dependency parsers in multiple languages and shows faster parsing speed. It is interesting to see that our greedy parser outperformed most other greedy dependency parsers. This is because our parser used both bootstrapping and Zhang and Nivre (2011)’s non-local features, which had not been used by other greedy parsers.

In the future, we will experiment with more advanced dependency representations (de Marneffe and Manning, 2008; Choi and Palmer, 2012b) to show robustness of our approach. Furthermore, we will evaluate individual methods of our approach separately to show impact of each method on parsing performance. We also plan to implement the typical beam search approach to make a direct comparison to our selectional branching.<sup>8</sup>

## Acknowledgments

Special thanks are due to Luke Vilnis of the University of Massachusetts Amherst for insights on

<sup>8</sup>Our parser is publicly available under an open source project, ClearNLP ([clearnlp.googlecode.com](http://clearnlp.googlecode.com)).

the ADAGRAD derivation. We gratefully acknowledge a grant from the Defense Advanced Research Projects Agency (DARPA) under the DEFT project, solicitation #: DARPA-BAA-12-47.

## References

- Bernd Bohnet and Joakim Nivre. 2012. A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP’12, pages 1455–1465.
- Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL’06, pages 149–164.
- Xavier Carreras, Michael Collins, and Terry Koo. 2008. TAG, Dynamic Programming, and the Perceptron for Efficient, Feature-rich Parsing. In *Proceedings of the 12th Conference on Computational Natural Language Learning*, CoNLL’08, pages 9–16.
- Daniel Cer, Marie-Catherine de Marneffe, Daniel Jurafsky, and Christopher D. Manning. 2010. Parsing to Stanford Dependencies: Trade-offs between speed and accuracy. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, LREC’10.
- Jinho D. Choi and Martha Palmer. 2011. Getting the Most out of Transition-based Dependency Parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL:HLT’11, pages 687–692.
- Jinho D. Choi and Martha Palmer. 2012a. Fast and Robust Part-of-Speech Tagging Using Dynamic Model Selection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, ACL’12, pages 363–367.
- Jinho D. Choi and Martha Palmer. 2012b. Guidelines for the Clear Style Constituent to Dependency Conversion. Technical Report 01-12, University of Colorado Boulder.
- Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the conference on Empirical methods in natural language processing*, EMNLP’02, pages 1–8.
- Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Proceedings of the COLING workshop on Cross-Framework and Cross-Domain Parser Evaluation*.

- Paramveer S. Dhillon, Jordan Rodu, Michael Collins, Dean P. Foster, and Lyle H. Ungar. 2012. Spectral Dependency Parsing with Latent Variables. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'12, pages 205–213.
- John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *The Journal of Machine Learning Research*, 12(39):2121–2159.
- Daniel Fernández-González and Carlos Gómez-Rodríguez. 2012. Improving Transition-Based Dependency Parsing with Buffer Transitions. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP'12, pages 308–319.
- Yoav Goldberg and Michael Elhadad. 2010. An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT:NAACL'10, pages 742–750.
- Yoav Goldberg and Joakim Nivre. 2012. A Dynamic Oracle for Arc-Eager Dependency Parsing. In *Proceedings of the 24th International Conference on Computational Linguistics*, COLING'12.
- Johan Hall, Joakim Nivre, and Jens Nilsson. 2006. Discriminative Classifiers for Deterministic Dependency Parsing. In *In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, COLING-ACL'06, pages 316–323.
- Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathya Keerthi, and S. Sundararajan. 2008. A Dual Coordinate Descent Method for Large-scale Linear SVM. In *Proceedings of the 25th international conference on Machine learning*, ICML'08, pages 408–415.
- Liang Huang and Kenji Sagae. 2010. Dynamic Programming for Linear-Time Incremental Parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL'10.
- Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured Perceptron with Inexact Search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT'12, pages 142–151.
- Terry Koo and Michael Collins. 2010. Efficient Third-order Dependency Parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL'10.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple Semi-supervised Dependency Parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, ACL:HLT'08, pages 595–603.
- Sandra Kübler, Ryan T. McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.
- Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- André F. T. Martins, Noah A. Smith, Eric P. Xing, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2010. Turbo Parsers: Dependency Parsing by Approximate Variational Inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP'10, pages 34–44.
- Ryan Mcdonald and Fernando Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithms. In *Proceedings of the Annual Meeting of the European American Chapter of the Association for Computational Linguistics*, EACL'06, pages 81–88.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online Large-Margin Training of Dependency Parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 91–98.
- Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual Dependency Analysis with a Two-Stage Discriminative Parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL'06, pages 216–220.
- Joakim Nivre and Ryan McDonald. 2008. Integrating Graph-based and Transition-based Dependency Parsers. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL:HLT'08, pages 950–958.
- Joakim Nivre and Jens Nilsson. 2005. Pseudo-Projective Dependency Parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL'05, pages 99–106.
- Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning*, CoNLL'06, pages 221–225.
- Joakim Nivre. 2003. An Efficient Algorithm for Projective Dependency Parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies*, IWPT'03, pages 149–160.

- Joakim Nivre. 2004. Incrementality in Deterministic Dependency Parsing. In *Proceedings of the ACL'04 Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57.
- Joakim Nivre. 2006. *Inductive Dependency Parsing*. Springer.
- Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.
- Joakim Nivre. 2009. Non-Projective Dependency Parsing in Expected Linear Time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, ACL-IJCNLP'09*, pages 351–359.
- Alexander M. Rush and Slav Petrov. 2012. Vine Pruning for Efficient Multi-Pass Dependency Parsing. In *Proceedings of the 12th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL:HLT'12*.
- Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On Dual Decomposition and Linear Programming Relaxations for Natural Language Processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP'10*, pages 1–11.
- Kenji Sagae and Alon Lavie. 2006. Parser Combination by Reparsing. In *In Proceedings of the Human Language Technology Conference of the NAACL, NAACL'06*, pages 129–132.
- Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An Empirical Study of Semi-supervised Structured Conditional Models for Dependency Parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP'09*, pages 551–560.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machine. In *Proceedings of the 8th International Workshop on Parsing Technologies, IWPT'03*, pages 195–206.
- Yue Zhang and Stephen Clark. 2008. A Tale of Two Parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP'08*, pages 562–571.
- Hao Zhang and Ryan McDonald. 2012. Generalized Higher-Order Dependency Parsing with Cube Pruning. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL'12*, pages 320–331.
- Yue Zhang and Joakim Nivre. 2011. Transition-based Dependency Parsing with Rich Non-local Features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL'11*, pages 188–193.

# Bilingually-Guided Monolingual Dependency Grammar Induction

Kai Liu<sup>†§</sup>, Yajuan Lü<sup>†</sup>, Wenbin Jiang<sup>†</sup>, Qun Liu<sup>†‡</sup>

<sup>†</sup>Key Laboratory of Intelligent Information Processing  
Institute of Computing Technology, Chinese Academy of Sciences  
P.O. Box 2704, Beijing 100190, China

{liukai,lvajuan,jiangwenbin,liuqun}@ict.ac.cn

<sup>‡</sup>Centre for Next Generation Localisation

Faculty of Engineering and Computing, Dublin City University

qliu@computing.dcu.ie

<sup>§</sup>University of Chinese Academy of Sciences

## Abstract

This paper describes a novel strategy for automatic induction of a monolingual dependency grammar under the guidance of bilingually-projected dependency. By moderately leveraging the dependency information projected from the parsed counterpart language, and simultaneously mining the underlying syntactic structure of the language considered, it effectively integrates the advantages of bilingual projection and unsupervised induction, so as to induce a monolingual grammar much better than previous models only using bilingual projection or unsupervised induction. We induced dependency grammar for five different languages under the guidance of dependency information projected from the parsed English translation, experiments show that the bilingually-guided method achieves a significant improvement of 28.5% over the unsupervised baseline and 3.0% over the best projection baseline on average.

## 1 Introduction

In past decades supervised methods achieved the state-of-the-art in constituency parsing (Collins, 2003; Charniak and Johnson, 2005; Petrov et al., 2006) and dependency parsing (McDonald et al., 2005a; McDonald et al., 2006; Nivre et al., 2006; Nivre et al., 2007; Koo and Collins, 2010). For supervised models, the human-annotated corpora on which models are trained, however, are expensive and difficult to build. As alternative strategies, methods which utilize raw texts have been investigated recently, including unsupervised meth-

ods which use only raw texts (Klein and Manning, 2004; Smith and Eisner, 2005; William et al., 2009), and semi-supervised methods (Koo et al., 2008) which use both raw texts and annotated corpus. And there are a lot of efforts have also been devoted to bilingual projection (Chen et al., 2010), which resorts to bilingual text with one language parsed, and projects the syntactic information from the parsed language to the unparsed one (Hwa et al., 2005; Ganchev et al., 2009).

In dependency grammar induction, unsupervised methods achieve continuous improvements in recent years (Klein and Manning, 2004; Smith and Eisner, 2005; Bod, 2006; William et al., 2009; Spitkovsky et al., 2010). Relying on a predefined distributional assumption and iteratively maximizing an approximate indicator (entropy, likelihood, etc.), an unsupervised model usually suffers from two drawbacks, i.e., lower performance and higher computational cost. On the contrary, bilingual projection (Hwa et al., 2005; Smith and Eisner, 2009; Jiang and Liu, 2010) seems a promising substitute for languages with a large amount of bilingual sentences and an existing parser of the counterpart language. By projecting syntactic structures directly (Hwa et al., 2005; Smith and Eisner, 2009; Jiang and Liu, 2010) across bilingual texts or indirectly across multilingual texts (Snyder et al., 2009; McDonald et al., 2011; Naseem et al., 2012), a better dependency grammar can be easily induced, if syntactic isomorphism is largely maintained between target and source languages.

Unsupervised induction and bilingual projection run according to totally different principles, the former mines the underlying structure of the monolingual language, while the latter leverages the syntactic knowledge of the parsed counter-

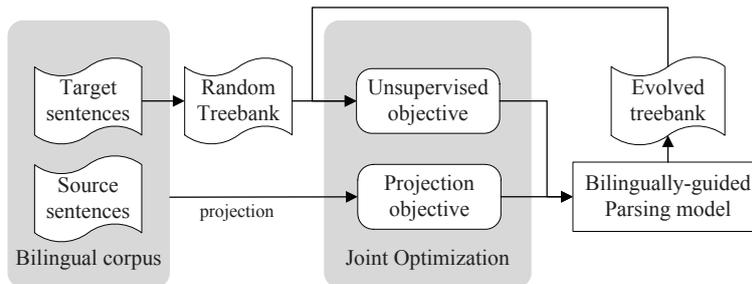


Figure 1: Training the bilingually-guided parsing model by iteration.

part language. Considering this, we propose a novel strategy for automatically inducing a monolingual dependency grammar under the guidance of bilingually-projected dependency information, which integrates the advantage of bilingual projection into the unsupervised framework. A randomly-initialized monolingual treebank evolves in a self-training iterative procedure, and the grammar parameters are tuned to simultaneously maximize both the monolingual likelihood and bilingually-projected likelihood of the evolving treebank. The monolingual likelihood is similar to the optimization objectives of conventional unsupervised models, while the bilingually-projected likelihood is the product of the projected probabilities of dependency trees. By moderately leveraging the dependency information projected from the parsed counterpart language, and simultaneously mining the underlying syntactic structure of the language considered, we can automatically induce a monolingual dependency grammar which is much better than previous models only using bilingual projection or unsupervised induction. In addition, since both likelihoods are fundamentally factorized into dependency edges (of the hypothesis tree), the computational complexity approaches to unsupervised models, while with much faster convergence. We evaluate the final automatically-induced dependency parsing model on 5 languages. Experimental results show that our method significantly outperforms previous work based on unsupervised method or indirect/direct dependency projection, where we see an average improvement of 28.5% over unsupervised baseline on all languages, and the improvements are 3.9%/3.0% over indirect/direct baselines. And our model achieves the most significant gains on Chinese, where the improvements are 12.0%, 4.5% over indirect and direct projection baselines respectively.

In the rest of the paper, we first describe the unsupervised dependency grammar induction framework in section 2 (where the unsupervised optimization objective is given), and introduce the bilingual projection method for dependency parsing in section 3 (where the projected optimization objective is given); Then in section 4 we present the bilingually-guided induction strategy for dependency grammar (where the two objectives above are jointly optimized, as shown in Figure 1). After giving a brief introduction of previous work in section 5, we finally give the experimental results in section 6 and conclude our work in section 7.

## 2 Unsupervised Dependency Grammar Induction

In this section, we introduce the unsupervised objective and the unsupervised training algorithm which is used as the framework of our bilingually-guided method. Unlike previous unsupervised work (Klein and Manning, 2004; Smith and Eisner, 2005; Bod, 2006), we select a self-training approach (similar to hard EM method) to train the unsupervised model. And the framework of our unsupervised model builds a random treebank on the monolingual corpus firstly for initialization and trains a discriminative parsing model on it. Then we use the parser to build an evolved treebank with the 1-best result for the next iteration run. In this way, the parser and treebank evolve in an iterative way until convergence. Let's introduce the parsing objective firstly:

Define  $e_i$  as the  $i^{th}$  word in monolingual sentence  $E$ ;  $d_{e_{ij}}$  denotes the word pair dependency relationship ( $e_i \rightarrow e_j$ ). Based on the features around  $d_{e_{ij}}$ , we can calculate the probability  $Pr(y|d_{e_{ij}})$  that the word pair  $d_{e_{ij}}$  can form a dependency arc

as:

$$Pr(y|d_{e_{ij}}) = \frac{1}{Z(d_{e_{ij}})} \exp\left(\sum_n \lambda_n \cdot f_n(d_{e_{ij}}, y)\right) \quad (1)$$

where  $y$  is the category of the relationship of  $d_{e_{ij}}$ :  $y = +$  means it is the probability that the word pair  $d_{e_{ij}}$  can form a dependency arc and  $y = -$  means the contrary.  $\lambda_n$  denotes the weight for feature function  $f_n(d_{e_{ij}}, y)$ , and the features we used are presented in Table 1 (Section 6).  $Z(d_{e_{ij}})$  is a normalizing constant:

$$Z(d_{e_{ij}}) = \sum_y \exp\left(\sum_n \lambda_n \cdot f_n(d_{e_{ij}}, y)\right) \quad (2)$$

Given a sentence  $E$ , parsing a dependency tree is to find a dependency tree  $D_E$  with maximum probability  $P_E$ :

$$P_E = \arg \max_{D_E} \prod_{d_{e_{ij}} \in D_E} Pr(+|d_{e_{ij}}) \quad (3)$$

## 2.1 Unsupervised Objective

We select a simple classifier objective function as the unsupervised objective function which is instinctively in accordance with the parsing objective:

$$\theta(\lambda) = \prod_{d_e \in D_{\bar{E}}} Pr(+|d_e) \prod_{d_e \in \tilde{D}_{\bar{E}}} Pr(-|d_e) \quad (4)$$

where  $\bar{E}$  is the monolingual corpus and  $E \in \bar{E}$ ,  $D_{\bar{E}}$  is the treebank that contains all  $D_E$  in the corpus, and  $\tilde{D}_{\bar{E}}$  denotes all other possible dependency arcs which do not exist in the treebank.

Maximizing the Formula (4) is equivalent to maximizing the following formula:

$$\begin{aligned} \theta_1(\lambda) = & \sum_{d_e \in D_{\bar{E}}} \log Pr(+|d_e) \\ & + \sum_{d_e \in \tilde{D}_{\bar{E}}} \log Pr(-|d_e) \end{aligned} \quad (5)$$

Since the size of edges between  $D_{\bar{E}}$  and  $\tilde{D}_{\bar{E}}$  is disproportionate, we use an empirical value to reduce the impact of the huge number of negative instances:

$$\begin{aligned} \theta_2(\lambda) = & \sum_{d_e \in D_{\bar{E}}} \log Pr(+|d_e) \\ & + \frac{|D_{\bar{E}}|}{|\tilde{D}_{\bar{E}}|} \sum_{d_e \in \tilde{D}_{\bar{E}}} \log Pr(-|d_e) \end{aligned} \quad (6)$$

where  $|x|$  is the size of  $x$ .

## Algorithm 1 Training unsupervised model

---

```

1: build random  $D_{\bar{E}}$ 
2:  $\lambda \leftarrow \text{train}(D_{\bar{E}}, \tilde{D}_{\bar{E}})$ 
3: repeat
4:   for each  $E \in \bar{E}$  do ▷ E step
5:      $D_E \leftarrow \text{parse}(E, \lambda)$ 
6:      $\lambda \leftarrow \text{train}(D_{\bar{E}}, \tilde{D}_{\bar{E}})$  ▷ M step
7: until convergence

```

---

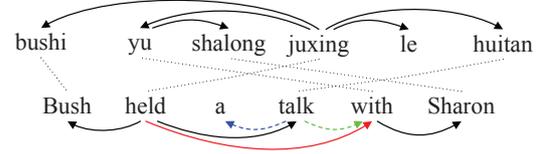


Figure 2: Projecting a Chinese dependency tree to English side according to DPA. Solid arrows are projected dependency arcs; dashed arrows are missing dependency arcs.

## 2.2 Unsupervised Training Algorithm

Algorithm 1 outlines the unsupervised training in its entirety, where the treebank  $D_{\bar{E}}$  and unsupervised parsing model with  $\lambda$  are updated iteratively.

In line 1 we build a random treebank  $D_{\bar{E}}$  on the monolingual corpus, and then train the parsing model with it (line 2) through a training procedure  $\text{train}(\cdot, \cdot)$  which needs  $D_{\bar{E}}$  and  $\tilde{D}_{\bar{E}}$  as classification instances. From line 3-7, we train the unsupervised model in self training iterative procedure, where line 4-5 are similar to the E-step in EM algorithm where calculates objective instead of expectation of 1-best tree (line 5) which is parsed according to the parsing objective (Formula 3) by parsing process  $\text{parse}(\cdot, \cdot)$ , and update the tree bank with the tree. Similar to M-step in EM, the algorithm maximizes the whole treebank's unsupervised objective (Formula 6) through the training procedure (line 6).

## 3 Bilingual Projection of Dependency Grammar

In this section, we introduce our projection objective and training algorithm which trains the model with arc instances.

Because of the heterogeneity between different languages and word alignment errors, projection methods may contain a lot of noises. Take Figure 2 as an example, following the Direct Projection Algorithm (DPA) (Hwa et al., 2005) (Section 5), the dependency relationships between words can be directly projected from the source

---

**Algorithm 2** Training projection model

---

```
1:  $D_P, D_N \leftarrow \text{proj}(\overline{F}, D_{\overline{F}}, A, \overline{E})$ 
2: repeat  $\triangleright \text{train}(D_P, D_N)$ 
3:    $\nabla\phi \leftarrow \text{grad}(D_P, D_N, \phi(\lambda))$ 
4:    $\lambda \leftarrow \text{climb}(\phi, \nabla\phi, \lambda)$ 
5: until maximization
```

---

language to the target language. Therefore, we can hardly obtain a treebank with complete trees through direct projection. So we extract projected discrete dependency arc instances instead of treebank as training set for the projected grammar induction model.

### 3.1 Projection Objective

Correspondingly, we select an objective which has the same form with the unsupervised one:

$$\begin{aligned} \phi(\lambda) = & \sum_{d_e \in D_P} \log Pr(+|d_e) \\ & + \sum_{d_e \in D_N} \log Pr(-|d_e) \end{aligned} \quad (7)$$

where  $D_P$  is the positive dependency arc instance set, which is obtained by direct projection methods (Hwa et al., 2005; Jiang and Liu, 2010) and  $D_N$  is the negative one.

### 3.2 Projection Algorithm

Basically, the training procedure in line 2,7 of Algorithm 1 can be divided into smaller iterative steps, and Algorithm 2 outlines the training step of projection model with instances.  $\overline{F}$  in Algorithm 2 is source sentences in bilingual corpus, and  $A$  is the alignments. Function  $\text{grad}(\cdot, \cdot, \cdot)$  gives the gradient ( $\nabla\phi$ ) and the objective is optimized with a generic optimization step (such as an LBFGS iteration (Zhu et al., 1997)) in the subroutine  $\text{climb}(\cdot, \cdot, \cdot)$ .

## 4 Bilingually-Guided Dependency Grammar Induction

This section presents our bilingually-guided grammar induction model, which incorporates unsupervised framework and bilingual projection model through a joint approach.

According to following observation: unsupervised induction model mines underlying syntactic structure of the monolingual language, however, it is hard to find good grammar induction in the exponential parsing space; bilingual projection obtains relatively reliable syntactic knowledge of the

parsed counterpart, but it possibly contains a lot of noises (e.g. Figure 2). We believe that unsupervised model and projection model can complement each other and a joint model which takes better use of both unsupervised parse trees and projected dependency arcs can give us a better parser.

Based on the idea, we propose a novel strategy for training monolingual grammar induction model with the guidance of unsupervised and bilingually-projected dependency information. Figure 1 outlines our bilingual-guided grammar induction process in its entirety. In our method, we select compatible objectives for unsupervised and projection models, in order to they can share the same grammar parameters. Then we incorporate projection model into our iterative unsupervised framework, and jointly optimize unsupervised and projection objectives with evolving treebank and constant projection information respectively. In this way, our bilingually-guided model’s parameters are tuned to simultaneously maximizing both monolingual likelihood and bilingually-projected likelihood by 4 steps:

1. Randomly build treebank on target sentences for initialization, and get the projected arc instances through projection from bitext.
2. Train the bilingually-guided grammar induction model by multi-objective optimization method with unsupervised objective and projection objective on treebank and projected arc instances respectively.
3. Use the parsing model to build new treebank on target language for next iteration.
4. Repeat steps 1, 2 and 3 until convergence.

The unsupervised objective is optimized by the loop—”tree bank→optimized model→new tree bank”. The treebank is evolved for runs. The unsupervised model gets projection constraint implicitly from those parse trees which contain information from projection part. The projection objective is optimized by the circulation—”projected instances→optimized model”, these projected instances will not change once we get them.

The iterative procedure proposed here is not a co-training algorithm (Sarkar, 2001; Hwa et al., 2003), because the input of the projection objective is static.

## 4.1 Joint Objective

For multi-objective optimization method, we employ the classical weighted-sum approach which just calculates the weighted linear sum of the objectives:

$$OBJ = \sum_m weight_m obj_m \quad (8)$$

We combine the unsupervised objective (Formula (6)) and projection objective (Formula (7)) together through the weighted-sum approach in Formula (8):

$$\ell(\lambda) = \alpha\theta_2(\lambda) + (1 - \alpha)\phi(\lambda) \quad (9)$$

where  $\ell(\lambda)$  is our weight-sum objective. And  $\alpha$  is a mixing coefficient which reflects the relative confidence between the unsupervised and projection objectives. Equally,  $\alpha$  and  $(1 - \alpha)$  can be seen as the weights in Formula (8). In that case, we can use a single parameter  $\alpha$  to control both weights for different objective functions. When  $\alpha = 1$  it is the unsupervised objective function in Formula (6). Contrary, if  $\alpha = 0$ , it is the projection objective function (Formula (7)) for projected instances.

With this approach, we can optimize the mixed parsing model by maximizing the objective in Formula (9). Though the function (Formula (9)) is an interpolation function, we use it for training instead of parsing. In the parsing procedure, our method calculates the probability of a dependency arc according to the Formula (2), while the interpolating method calculates it by:

$$Pr(y|d_{e_{ij}}) = \alpha Pr_1(y|d_{e_{ij}}) + (1 - \alpha) Pr_2(y|d_{e_{ij}}) \quad (10)$$

where  $Pr_1(y|d_{e_{ij}})$  and  $Pr_2(y|d_{e_{ij}})$  are the probabilities provided by different models.

## 4.2 Training Algorithm

We optimize the objective (Formula (9)) via a gradient-based search algorithm. And the gradient with respect to  $\lambda_k$  takes the form:

$$\nabla\ell(\lambda_k) = \alpha \frac{\partial\theta_2(\lambda)}{\partial\lambda_k} + (1 - \alpha) \frac{\partial\phi(\lambda)}{\partial\lambda_k} \quad (11)$$

Algorithm 3 outlines our joint training procedure, which tunes the grammar parameter  $\lambda$  simultaneously maximize both unsupervised objective

---

### Algorithm 3 Training joint model

---

```

1:  $D_P, D_N \leftarrow proj(\overline{F}, D_{\overline{F}}, A, \overline{E})$ 
2: build random  $D_{\overline{E}}$ 
3:  $\lambda \leftarrow train(D_P, D_N)$ 
4: repeat
5:   for each  $E \in \overline{E}$  do ▷ E step
6:      $D_{\overline{E}} \leftarrow parse(E, \lambda)$ 
7:      $\nabla\ell(\lambda) \leftarrow grad(D_{\overline{E}}, \tilde{D}_{\overline{E}}, D_P, D_N, \ell(\lambda))$ 
8:      $\lambda \leftarrow climb(\ell(\lambda), \nabla\ell(\lambda), \lambda)$  ▷ M step
9: until convergence

```

---

and projection objective. And it incorporates unsupervised framework and projection model algorithm together. It is grounded on the work which uses features in the unsupervised model (Berg-Kirkpatrick et al., 2010).

In line 1, 2 we get projected dependency instances from source side according to projection methods and build a random treebank (step 1). Then we train an initial model with projection instances in line 3. From line 4-9, the objective is optimized with a generic optimization step in the subroutine *climb*( $\cdot, \cdot, \cdot, \cdot, \cdot$ ). For each sentence we parse its dependency tree, and update the tree into the treebank (step 3). Then we calculate the gradient and optimize the joint objective according to the evolved treebank and projected instances (step 2). Lines 5-6 are equivalent to the E-step of the EM algorithm, and lines 7-8 are equivalent to the M-step.

## 5 Related work

The DMV (Klein and Manning, 2004) is a single-state head automata model (Alshawi, 1996) which is based on POS tags. And DMV learns the grammar via inside-outside re-estimation (Baker, 1979) without any smoothing, while Spitzkovsky et al. (2010) utilizes smoothing and learning strategy during grammar learning and William et al. (2009) improves DMV with richer context.

The dependency projection method DPA (Hwa et al., 2005) based on Direct Correspondence Assumption (Hwa et al., 2002) can be described as: if there is a pair of source words with a dependency relationship, the corresponding aligned words in target sentence can be considered as having the same dependency relationship equivalently (e.g. Figure 2). The Word Pair Classification (WPC) method (Jiang and Liu, 2010) modifies the DPA method and makes it more robust. Smith and Eisner (2009) propose an adaptation method founded on quasi-synchronous grammar features

Type	Feature Template		
Unigram	$word_i$	$pos_i$	$word_i \circ pos_i$
	$word_j$	$pos_j$	$word_j \circ pos_j$
Bigram	$word_i \circ pos_j$	$word_j \circ pos_i$	$pos_i \circ pos_j$
	$word_i \circ word_j$	$word_i \circ pos_i \circ word_j$	$word_i \circ word_j \circ pos_j$
	$word_i \circ pos_i \circ pos_j$	$pos_i \circ word_j \circ pos_j$	
	$word_i \circ pos_i \circ word_j \circ pos_j$		
Surrounding	$pos_{i-1} \circ pos_i \circ pos_j$	$pos_i \circ pos_{i+1} \circ pos_j$	$pos_i \circ pos_{j-1} \circ pos_j$
	$pos_i \circ pos_j \circ pos_{j+1}$	$pos_{i-1} \circ pos_i \circ pos_{j-1}$	$pos_i \circ pos_{i+1} \circ pos_{j+1}$
	$pos_{i-1} \circ pos_{j-1} \circ pos_j$	$pos_{i+1} \circ pos_j \circ pos_{j+1}$	$pos_{i-1} \circ pos_i \circ pos_{j+1}$
	$pos_i \circ pos_{i+1} \circ pos_{j-1}$	$pos_{i-1} \circ pos_j \circ pos_{j+1}$	$pos_{i+1} \circ pos_{j-1} \circ pos_j$
	$pos_{i-1} \circ pos_i \circ pos_{j-1} \circ pos_j$	$pos_i \circ pos_{i+1} \circ pos_j \circ pos_{j+1}$	
	$pos_i \circ pos_{i+1} \circ pos_{j-1} \circ pos_j$	$pos_{i-1} \circ pos_i \circ pos_j \circ pos_{j+1}$	

Table 1: Feature templates for dependency parsing. For edge  $d_{e_{ij}}$ :  $word_i$  is the parent word and  $word_j$  is the child word, similar to "pos". "+1" denotes the preceding token of the sentence, similar to "-1".

for dependency projection and annotation, which requires a small set of dependency annotated corpus of target language.

Similarly, using indirect information from multilingual (Cohen et al., 2011; Täckström et al., 2012) is an effective way to improve unsupervised parsing. (Zeman and Resnik, 2008; McDonald et al., 2011; Sjøgaard, 2011) employ non-lexicalized parser trained on other languages to process a target language. McDonald et al. (2011) adapts their multi-source parser according to DCA, while Naseem et al. (2012) selects a selective sharing model to make better use of grammar information in multi-sources.

Due to similar reasons, many works are devoted to POS projection (Yarowsky et al., 2001; Shen et al., 2007; Naseem et al., 2009), and they also suffer from similar problems. Some seek for unsupervised methods, e.g. Naseem et al. (2009), and some further improve the projection by a graph-based projection (Das and Petrov, 2011).

Our model differs from the approaches above in its emphasis on utilizing information from both sides of bilingual corpus in an unsupervised training framework, while most of the work above only utilize the information from a single side.

## 6 Experiments

In this section, we evaluate the performance of the MST dependency parser (McDonald et al., 2005b) which is trained by our bilingually-guided model on 5 languages. And the features used in our experiments are summarized in Table 1.

### 6.1 Experiment Setup

**Datasets and Evaluation** Our experiments are run on five different languages: Chinese(ch), Danish(da), Dutch(nl), Portuguese(pt) and

Swedish(sv) (da, nl, pt and sv are free data sets distributed for the 2006 CoNLL Shared Tasks (Buchholz and Marsi, 2006)). For all languages, we only use English-target parallel data: we take the FBIS English-Chinese bitext as bilingual corpus for English-Chinese dependency projection which contains 239K sentence pairs with about 8.9M/6.9M words in English/Chinese, and for other languages we use the readily available data in the Europarl corpus. Then we run tests on the Penn Chinese Treebank (CTB) and CoNLL-X test sets.

English sentences are tagged by the implementations of the POS tagger of Collins (2002), which is trained on WSJ. The source sentences are then parsed by an implementation of 2nd-ordered MST model of McDonald and Pereira (2006), which is trained on dependency trees extracted from Penn Treebank.

As the evaluation metric, we use parsing accuracy which is the percentage of the words which have found their correct parents. We evaluate on sentences with all length for our method.

**Training Regime** In experiments, we use the projection method proposed by Jiang and Liu (2010) to provide the projection instances. And we train the projection part  $\alpha = 0$  first for initialization, on which the whole model will be trained. Availing of the initialization method, the model can converge very fast (about 3 iterations is sufficient) and the results are more stable than the ones trained on random initialization.

**Baselines** We compare our method against three kinds of different approaches: unsupervised method (Klein and Manning, 2004); single-source direct projection methods (Hwa et al., 2005; Jiang and Liu, 2010); multi-source indirect projection methods with multi-sources (M-

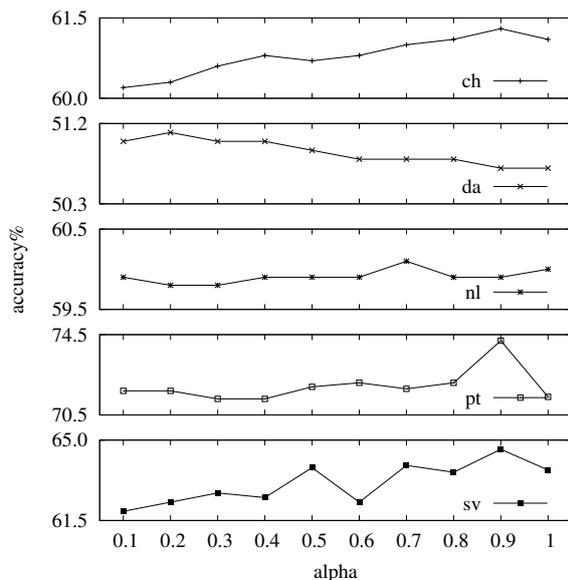


Figure 3: The performance of our model with respect to a series of ratio  $\alpha$

cDonald et al., 2011; Naseem et al., 2012).

## 6.2 Results

We test our method on CTB and CoNLL-X free test data sets respectively, and the performance is summarized in Table 2. Figure 3 presents the performance with different  $\alpha$  on different languages.

**Compare against Unsupervised Baseline** Experimental results show that our unsupervised framework’s performance approaches to the DMV method. And the bilingually-guided model can promote the unsupervised method consistency over all languages. On the best results’ average of four comparable languages (da, nl, pt, sv), the promotion gained by our model is 28.5% over the baseline method (DMV) (Klein and Manning, 2004).

**Compare against Projection Baselines** For all languages, the model consistently outperforms on direct projection baseline. On the average of each language’s best result, our model outperforms all kinds of baselines, yielding 3.0% gain over the single-source direct-projection method (Jiang and Liu, 2010) and 3.9% gain over the multi-source indirect-projection method (McDonald et al., 2011). On the average of all results with different parameters, our method also gains more than 2.0% improvements on all baselines. Particularly, our model achieves the most significant gains on Chinese, where the improvements are 4.5%/12.0% on direct/indirect projection base-

Model	Accuracy%					
	ch	da	nl	pt	sv	avg
DMV	42.5*	33.4	38.5	20.1	44.0	—.—
DPA	53.9	—.—	—.—	—.—	—.—	—.—
WPC	56.8	50.1	58.4	70.5	60.8	59.3
Transfer	49.3	49.5	53.9	75.8	63.6	58.4
Selective	51.2	—.—	55.9	73.5	61.5	—.—
<i>unsuper</i>	22.6	41.6	15.2	45.7	42.4	33.5
avg	61.0	50.7	59.9	72.0	63.1	61.3
max	61.3	51.1	60.1	74.2	64.6	62.3

Table 2: The directed dependency accuracy with different parameter of our model and the baselines. The first section of the table (row 3-7) shows the results of the baselines: a unsupervised method baseline (Klein and Manning, 2004)(DMV); a single-source projection method baseline (Hwa et al., 2005) (DPA) and its improvement (Jiang and Liu, 2010)(WPC); two multi-source baselines (McDonald et al., 2011)(Transfer) and (Naseem et al., 2012)(Selective). The second section of the table (row 8) presents the result of our unsupervised framework (unsuper). The third section gives the mean value (avg) and maximum value (max) of our model with different  $\alpha$  in Figure 3.

\*: The result is based on sentences with 10 words or less after the removal of punctuation, it is an incomparable result.

lines.

The results in Figure 3 prove that our unsupervised framework  $\alpha = 1$  can promote the grammar induction if it has a good start (well initialization), and it will be better once we incorporate the information from the projection side ( $\alpha = 0.9$ ). And the maximum points are not in  $\alpha = 1$ , which implies that projection information is still available for the unsupervised framework even if we employ the projection model as the initialization. So we suggest that a greater parameter is a better choice for our model. And there are some random factors in our model which make performance curves with more fluctuation. And there is just a little improvement shown in *da*, in which the same situation is observed by (McDonald et al., 2011).

## 6.3 Effects of the Size of Training Corpus

To investigate how the size of the training corpus influences the result, we train the model on extracted bilingual corpus with varying sizes: 10K, 50K, 100K, 150K and 200K sentences pairs.

As shown in Figure 4, our approach continu-

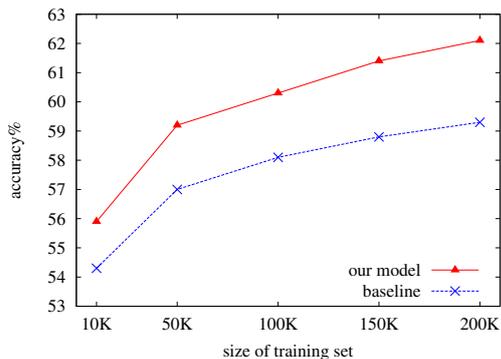


Figure 4: Performance on varying sizes (average of 5 languages,  $\alpha = 0.9$ )

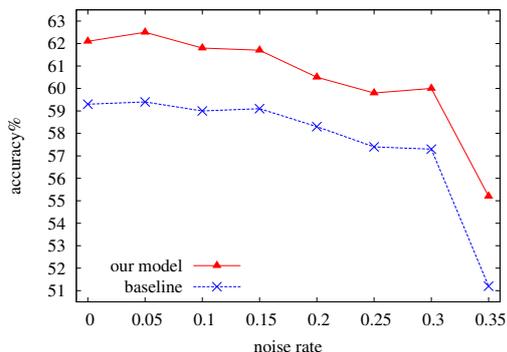


Figure 5: Performance on different projection quality (average of 5 languages,  $\alpha = 0.9$ ). The noise rate is the percentage of the projected instances being messed up.

ously outperforms the baseline with the increasing size of training corpus. It is especially noteworthy that the more training data is utilized the more superiority our model enjoys. That is, because our method not only utilizes the projection information but also avails itself of the monolingual corpus.

#### 6.4 Effect of Projection Quality

The projection quality can be influenced by the quality of the source parsing, alignments, projection methods, corpus quality and many other factors. In order to detect the effects of varying projection qualities on our approach, we simulate the complex projection procedure by messing up the projected instances randomly with different noise rates. The curves in Figure 5 show the performance of WPC baseline and our bilingual-guided method. For different noise rates, our model's results consistently outperform the baselines. When the noise rate is greater than 0.2, our improvement

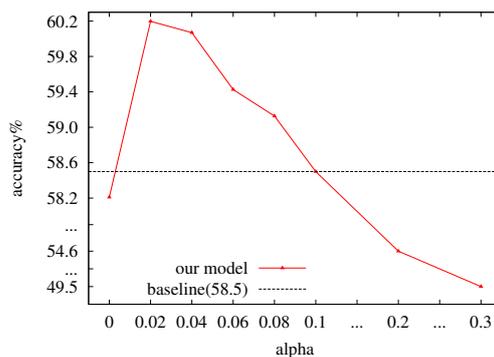


Figure 6: The performance curve of our model (random initialization) on Chinese, with respect to a series of ratio  $\alpha$ . The baseline is the result of WPC model.

increases with the growth of the noise rate. The result suggests that our method can solve some problems which are caused by projection noise.

#### 6.5 Performance on Random Initialization

We test our model with random initialization on different  $\alpha$ . The curve in Figure 6 shows the performance of our model on Chinese.

The results seem supporting our unsupervised optimization method when  $\alpha$  is in the range of  $(0, 0.1)$ . It implies that the unsupervised structure information is useful, but it seems creating a negative effect on the model when  $\alpha$  is greater than 0.1. Because the unsupervised part can gain constraints from the projection part. But with the increase of  $\alpha$ , the strength of constraint dwindles, and the unsupervised part will gradually lose control. And bad unsupervised part pulls the full model down.

### 7 Conclusion and Future Work

This paper presents a bilingually-guided strategy for automatic dependency grammar induction, which adopts an unsupervised skeleton and leverages the bilingually-projected dependency information during optimization. By simultaneously maximizing the monolingual likelihood and bilingually-projected likelihood in the EM procedure, it effectively integrates the advantages of bilingual projection and unsupervised induction. Experiments on 5 languages show that the novel strategy significantly outperforms previous unsupervised or bilingually-projected models. Since its computational complexity approaches to the skeleton unsupervised model (with much fewer iterations), and the bilingual text aligned to

resource-rich languages is easy to obtain, such a hybrid method seems to be a better choice for automatic grammar induction. It also indicates that the combination of bilingual constraint and unsupervised methodology has a promising prospect for grammar induction. In the future work we will investigate such kind of strategies, such as bilinearly unsupervised induction.

## Acknowledgments

The authors were supported by National Natural Science Foundation of China, Contracts 61202216, 863 State Key Project (No. 2011AA01A207), and National Key Technology R&D Program (No. 2012BAH39B03), Key Project of Knowledge Innovation Program of Chinese Academy of Sciences (No. KGZD-EW-501). Qun Liu's work is partially supported by Science Foundation Ireland (Grant No.07/CE/I1142) as part of the CNGL at Dublin City University. We would like to thank the anonymous reviewers for their insightful comments and those who helped to modify the paper.

## References

- H. Alshawi. 1996. Head automata for speech translation. In *Proc. of ICSLP*.
- James K Baker. 1979. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America*, 65:S132.
- T. Berg-Kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein. 2010. Painless unsupervised learning with features. In *HLT: NAACL*, pages 582–590.
- Rens Bod. 2006. An all-subtrees approach to unsupervised parsing. In *Proc. of the 21st ICCL and the 44th ACL*, pages 865–872.
- S. Buchholz and E. Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proc. of the 2002 Conference on EMNLP*. Proc. CoNLL.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proc. of the 43rd ACL*, pages 173–180, Ann Arbor, Michigan, June.
- W. Chen, J. Kazama, and K. Torisawa. 2010. Bilingual dependency parsing with bilingual subtree constraints. In *Proc. of ACL*, pages 21–29.
- S.B. Cohen, D. Das, and N.A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proc. of the Conference on EMNLP*, pages 50–61.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. of the 2002 Conference on EMNLP*, pages 1–8, July.
- Michael Collins. 2003. Head-driven statistical models for natural language parsing. In *Computational Linguistics*.
- D. Das and S. Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proc. of ACL*.
- K. Ganchev, J. Gillenwater, and B. Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proc. of IJCNLP of the AFNLP: Volume 1-Volume 1*, pages 369–377.
- R. Hwa, P. Resnik, A. Weinberg, and O. Kolak. 2002. Evaluating translational correspondence using annotation projection. In *Proc. of ACL*, pages 392–399.
- R. Hwa, M. Osborne, A. Sarkar, and M. Steedman. 2003. Corrected co-training for statistical parsers. In *ICML-03 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, Washington DC*.
- R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural language engineering*, 11(3):311–325.
- W. Jiang and Q. Liu. 2010. Dependency parsing and projection based on word-pair classification. In *Proc. of ACL*, pages 12–20.
- D. Klein and C.D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*, page 478.
- Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proc. of the 48th ACL*, pages 1–11, July.
- T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. pages 595–603.
- R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proc. of the 11th Conf. of EACL*.
- R. McDonald, K. Crammer, and F. Pereira. 2005a. Online large-margin training of dependency parsers. In *Proc. of ACL*, pages 91–98.
- R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of EMNLP*, pages 523–530.
- R. McDonald, K. Lerman, and F. Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proc. of CoNLL*, pages 216–220.

- R. McDonald, S. Petrov, and K. Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proc. of EMNLP*, pages 62–72. ACL.
- T. Naseem, B. Snyder, J. Eisenstein, and R. Barzilay. 2009. Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research*, 36(1):341–385.
- Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proc. of the 50th ACL*, pages 629–637, July.
- J. Nivre, J. Hall, J. Nilsson, G. Eryiğit, and S. Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proc. of CoNLL*, pages 221–225.
- J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. 2007. Malt-parser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. of the 21st ICCL & 44th ACL*, pages 433–440, July.
- A. Sarkar. 2001. Applying co-training methods to statistical parsing. In *Proc. of NAACL*, pages 1–8.
- L. Shen, G. Satta, and A. Joshi. 2007. Guided learning for bidirectional sequence classification. In *Annual Meeting-*, volume 45, page 760.
- N.A. Smith and J. Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. of ACL*, pages 354–362.
- D.A. Smith and J. Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proc. of EMNLP: Volume 2-Volume 2*, pages 822–831.
- B. Snyder, T. Naseem, and R. Barzilay. 2009. Unsupervised multilingual grammar induction. In *Proc. of IJCNLP of the AFNLP: Volume 1-Volume 1*, pages 73–81.
- Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *Proc. of the 49th ACL: HLT*, pages 682–686.
- Valentin I. Spitzkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2010. From baby steps to leapfrog: How “less is more” in unsupervised dependency parsing. In *HLT: NAACL*, pages 751–759, June.
- O. Täckström, R. McDonald, and J. Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure.
- William, M. Johnson, and D. McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proc. of NAACL*, pages 101–109.
- D. Yarowsky, G. Ngai, and R. Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proc. of HLT*, pages 1–8.
- Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proc. of the IJCNLP-08*. Proc. CoNLL.
- Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. 1997. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560.

# Joint Word Alignment and Bilingual Named Entity Recognition Using Dual Decomposition

**Mengqiu Wang**  
Stanford University  
Stanford, CA 94305  
mengqiu@cs.stanford.edu

**Wanxiang Che**  
Harbin Institute of Technology  
Harbin, China, 150001  
car@ir.hit.edu.cn

**Christopher D. Manning**  
Stanford University  
Stanford, CA 94305  
manning@cs.stanford.edu

## Abstract

Translated bi-texts contain complementary language cues, and previous work on Named Entity Recognition (NER) has demonstrated improvements in performance over monolingual taggers by promoting agreement of tagging decisions between the two languages. However, most previous approaches to bilingual tagging assume word alignments are given as fixed input, which can cause cascading errors. We observe that NER label information can be used to correct alignment mistakes, and present a graphical model that performs bilingual NER tagging jointly with word alignment, by combining two monolingual tagging models with two unidirectional alignment models. We introduce additional cross-lingual edge factors that encourage agreements between tagging and alignment decisions. We design a dual decomposition inference algorithm to perform joint decoding over the combined alignment and NER output space. Experiments on the OntoNotes dataset demonstrate that our method yields significant improvements in both NER and word alignment over state-of-the-art monolingual baselines.

## 1 Introduction

We study the problem of Named Entity Recognition (NER) in a bilingual context, where the goal is to annotate parallel bi-texts with named entity tags. This is a particularly important problem for machine translation (MT) since entities such as person names, locations, organizations, etc. carry much of the information expressed in the source

sentence. Recognizing them provides useful information for phrase detection and word sense disambiguation (e.g., “melody” as in a female name has a different translation from the word “melody” in a musical sense), and can be directly leveraged to improve translation quality (Babych and Hartley, 2003). We can also automatically construct a named entity translation lexicon by annotating and extracting entities from bi-texts, and use it to improve MT performance (Huang and Vogel, 2002; Al-Onaizan and Knight, 2002). Previous work such as Burkett et al. (2010b), Li et al. (2012) and Kim et al. (2012) have also demonstrated that bi-texts annotated with NER tags can provide useful additional training sources for improving the performance of standalone monolingual taggers.

Because human translation in general preserves semantic equivalence, bi-texts represent two perspectives on the same semantic content (Burkett et al., 2010b). As a result, we can find complementary cues in the two languages that help to disambiguate named entity mentions (Brown et al., 1991). For example, the English word “Jordan” can be either a last name or a country. Without sufficient context it can be difficult to distinguish the two; however, in Chinese, these two senses are disambiguated: “乔丹” as a last name, and “约旦” as a country name.

In this work, we first develop a bilingual NER model (denoted as BI-NER) by embedding two monolingual CRF-based NER models into a larger undirected graphical model, and introduce additional edge factors based on word alignment (WA). Because the new bilingual model contains many cyclic cliques, exact inference is intractable. We employ a dual decomposition (DD) inference algorithm (Bertsekas, 1999; Rush et al., 2010) for performing approximate inference. Unlike most

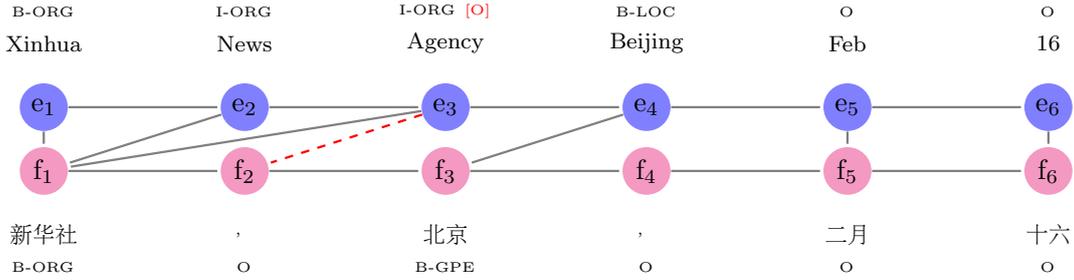


Figure 1: Example of NER labels between two word-aligned bilingual parallel sentences. The [O] tag is an example of a wrong tag assignment. The dashed alignment link between  $e_3$  and  $f_2$  is an example of alignment error.

previous applications of the DD method in NLP, where the model typically factors over two components and agreement is to be sought between the two (Rush et al., 2010; Koo et al., 2010; DeNero and Macherey, 2011; Chieu and Teow, 2012), our method decomposes the larger graphical model into many overlapping components where each alignment edge forms a separate factor. We design clique potentials over the alignment-based edges to encourage entity tag agreements. Our method does not require any manual annotation of word alignments or named entities over the bilingual training data.

The aforementioned BI-NER model assumes fixed alignment input given by an underlying word aligner. But the entity span and type predictions given by the NER models contain complementary information for correcting alignment errors. To capture this source of information, we present a novel extension that combines the BI-NER model with two uni-directional HMM-based alignment models, and perform joint decoding of NER and word alignments. The new model (denoted as BI-NER-WA) factors over five components: one NER model and one word alignment model for each language, plus a joint NER-alignment model which not only enforces NER label agreements but also facilitates message passing among the other four components. An extended DD decoding algorithm is again employed to perform approximate inference.

We give a formal definition of the Bi-NER model in Section 2, and then move to present the Bi-NER-WA model in Section 3.

## 2 Bilingual NER by Agreement

The inputs to our models are parallel sentence pairs (see Figure 1 for an example in English and

Chinese). We denote the sentences as  $e$  (for English) and  $f$  (for Chinese). We assume access to two monolingual linear-chain CRF-based NER models that are already trained. The English-side CRF model assigns the following probability for a tag sequence  $\mathbf{y}^e$ :

$$P_{CRF_e}(\mathbf{y}^e | \mathbf{e}) = \frac{\prod_{v_i \in \mathcal{V}^e} \psi(v_i) \prod_{(v_i, v_j) \in \mathcal{D}^e} \omega(v_i, v_j)}{\mathcal{Z}^e(\mathbf{e})}$$

where  $\mathcal{V}^e$  is the set of vertices in the CRF and  $\mathcal{D}^e$  is the set of edges.  $\psi(v_i)$  and  $\omega(v_i, v_j)$  are the node and edge clique potentials, and  $\mathcal{Z}^e(\mathbf{e})$  is the partition function for input sequence  $\mathbf{e}$  under the English CRF model. We let  $k(\mathbf{y}^e)$  be the un-normalized log-probability of tag sequence  $\mathbf{y}^e$ , defined as:

$$k(\mathbf{y}^e) = \log \left( \prod_{v_i \in \mathcal{V}^e} \psi(v_i) \prod_{(v_i, v_j) \in \mathcal{D}^e} \omega(v_i, v_j) \right)$$

Similarly, we define model  $P_{CRF_f}$  and un-normalized log-probability  $l(\mathbf{y}^f)$  for Chinese.

We also assume that a set of word alignments ( $\mathcal{A} = \{(i, j) : e_i \leftrightarrow f_j\}$ ) is given by a word aligner and remain fixed in our model.

For clarity, we assume  $\mathbf{y}^e$  and  $\mathbf{y}^f$  are binary variables in the description of our algorithms. The extension to the multi-class case is straight-forward and does not affect the core algorithms.

### 2.1 Hard Agreement

We define a BI-NER model which imposes hard agreement of entity labels over aligned word pairs. At inference time, we solve the following opti-

mization problem:

$$\begin{aligned}
& \max_{\mathbf{y}^e, \mathbf{y}^f} \log(P_{CRF_e}(\mathbf{y}^e)) + \log(P_{CRF_f}(\mathbf{y}^f)) \\
&= \max_{\mathbf{y}^e, \mathbf{y}^f} k(\mathbf{y}^e) + l(\mathbf{y}^f) - \log \mathcal{Z}_e(\mathbf{e}) - \log \mathcal{Z}_f(\mathbf{f}) \\
&\simeq \max_{\mathbf{y}^e, \mathbf{y}^f} k(\mathbf{y}^e) + l(\mathbf{y}^f) \\
&\quad \ni y_i^e = y_j^f \quad \forall (i, j) \in \mathcal{A}
\end{aligned}$$

We dropped the  $\mathcal{Z}_e(\mathbf{e})$  and  $\mathcal{Z}_f(\mathbf{f})$  terms because they remain constant at inference time.

The Lagrangian relaxation of this term is:

$$\begin{aligned}
L(\mathbf{y}^e, \mathbf{y}^f, \mathbf{U}) &= \\
& k(\mathbf{y}^e) + l(\mathbf{y}^f) + \sum_{(i,j) \in \mathcal{A}} u(i, j) (y_i^e - y_j^f)
\end{aligned}$$

where  $u(i, j)$  are the Lagrangian multipliers.

Instead of solving the Lagrangian directly, we can form the dual of this problem and solve it using dual decomposition (Rush et al., 2010):

$$\begin{aligned}
\min_{\mathbf{U}} \left( \max_{\mathbf{y}^e} \left[ k(\mathbf{y}^e) + \sum_{(i,j) \in \mathcal{A}} u(i, j) y_i^e \right] \right. \\
\left. + \max_{\mathbf{y}^f} \left[ l(\mathbf{y}^f) - \sum_{(i,j) \in \mathcal{A}} u(i, j) y_j^f \right] \right)
\end{aligned}$$

Similar to previous work, we solve this DD problem by iteratively updating the sub-gradient as depicted in Algorithm 1.  $T$  is the maximum number of iterations before early stopping, and  $\alpha_t$  is the learning rate at time  $t$ . We adopt a learning rate update rule from Koo et al. (2010) where  $\alpha_t$  is defined as  $\frac{1}{N}$ , where  $N$  is the number of times we observed a consecutive dual value increase from iteration 1 to  $t$ .

A thorough introduction to the theoretical foundations of dual decomposition algorithms is beyond the scope of this paper; we encourage unfamiliar readers to read Rush and Collins (2012) for a full tutorial.

## 2.2 Soft Agreement

The previously discussed hard agreement model rests on the core assumption that aligned words must have identical entity tags. In reality, however, this assumption does not always hold. Firstly, assuming words are correctly aligned, their entity tags may not agree due to inconsistency in annotation standards. In Figure 1, for example, the

---

### Algorithm 1 DD inference algorithm for hard agreement model.

---

```

 $\forall (i, j) \in \mathcal{A} : u(i, j) = 0$ 
for  $t \leftarrow 1$  to  $T$  do
   $\mathbf{y}^{e*} \leftarrow \operatorname{argmax}_{\mathbf{y}^e} k(\mathbf{y}^e) + \sum_{(i,j) \in \mathcal{A}} u(i, j) y_i^e$ 
   $\mathbf{y}^{f*} \leftarrow \operatorname{argmax}_{\mathbf{y}^f} l(\mathbf{y}^f) - \sum_{(i,j) \in \mathcal{A}} u(i, j) y_j^f$ 
  if  $\forall (i, j) \in \mathcal{A} : y_i^{e*} = y_j^{f*}$  then
    return  $(\mathbf{y}^{e*}, \mathbf{y}^{f*})$ 
  end if
  for all  $(i, j) \in \mathcal{A}$  do
     $u(i, j) \leftarrow u(i, j) + \alpha_t (y_j^{f*} - y_i^{e*})$ 
  end for
end for
return  $(\mathbf{y}_{(\mathbf{T})}^{e*}, \mathbf{y}_{(\mathbf{T})}^{f*})$ 

```

---

word ‘‘Beijing’’ can be either a Geo-Political Entity (GPE) or a location. The Chinese annotation standard may enforce that ‘‘Beijing’’ should always be tagged as GPE when it is mentioned in isolation, while the English standard may require the annotator to judge based on word usage context. The assumption in the hard agreement model can also be violated if there are word alignment errors.

In order to model this uncertainty, we extend the two previously independent CRF models into a larger undirected graphical model, by introducing a cross-lingual edge factor  $\phi(i, j)$  for every pair of word positions  $(i, j) \in \mathcal{A}$ . We associate a clique potential function  $h_{(i,j)}(y_i^e, y_j^f)$  for  $\phi(i, j)$ :

$$h_{(i,j)}(y_i^e, y_j^f) = \operatorname{pmi}(y_i^e, y_j^f)^{\hat{P}(e_i, f_j)}$$

where  $\operatorname{pmi}(y_i^e, y_j^f)$  is the point-wise mutual information (PMI) of the tag pair, and we raise it to the power of a posterior alignment probability  $\hat{P}(e_i, f_j)$ . For a pair of NEs that are aligned with low probability, we cannot be too sure about the association of the two NEs, therefore the model should not impose too much influence from the bilingual agreement model; instead, we will let the monolingual NE models make their decisions, and trust that those are the best estimates we can come up with when we do not have much confidence in their bilingual association. The use of the posterior alignment probability facilitates this purpose.

Initially, each of the cross-lingual edge factors will attempt to assign a pair of tags that has the highest PMI score, but if the monolingual taggers do not agree, a penalty will start accumulating over this pair, until some other pair that agrees better with the monolingual models takes the top spot.

Simultaneously, the monolingual models will also be encouraged to agree with the cross-lingual edge factors. This way, the various components effectively trade penalties indirectly through the cross-lingual edges, until a tag sequence that maximizes the joint probability is achieved.

Since we assume no bilingually annotated NER corpus is available, in order to get an estimate of the PMI scores, we first tag a collection of unannotated bilingual sentence pairs using the monolingual CRF taggers, and collect counts of aligned entity pairs from this auto-generated tagged data.

Each of the  $\phi(i, j)$  edge factors (e.g., the edge between node  $f_3$  and  $e_4$  in Figure 1) overlaps with each of the two CRF models over one vertex (e.g.,  $f_3$  on Chinese side and  $e_4$  on English side), and we seek agreement with the Chinese CRF model over tag assignment of  $f_j$ , and similarly for  $e_i$  on English side. In other words, no direct agreement between the two CRF models is enforced, but they both need to agree with the bilingual edge factors.

The updated optimization problem becomes:

$$\begin{aligned} & \max_{\mathbf{y}^{e(k)}, \mathbf{y}^{f(l)}, \mathbf{y}^{e(h)}, \mathbf{y}^{f(h)}} k \left( \mathbf{y}^{e(k)} \right) + l \left( \mathbf{y}^{f(l)} \right) + \\ & \sum_{(i,j) \in \mathcal{A}} h_{(i,j)} \left( y_i^{e(h)}, y_j^{f(h)} \right) \\ & \ni \forall (i, j) \in \mathcal{A}: \left( y_i^{e(k)} = y_i^{e(h)} \right) \wedge \left( y_j^{f(l)} = y_j^{f(h)} \right) \end{aligned}$$

where the notation  $y_i^{e(k)}$  denotes tag assignment to word  $e_i$  by the English CRF and  $y_i^{e(h)}$  denotes assignment to word  $e_i$  by the bilingual factor;  $y_j^{f(l)}$  denotes the tag assignment to word  $f_j$  by the Chinese CRF and  $y_j^{f(h)}$  denotes assignment to word  $f_j$  by the bilingual factor.

The updated DD algorithm is illustrated in Algorithm 2 (case 2). We introduce two separate sets of dual constraints  $\mathbf{w}^e$  and  $\mathbf{w}^f$ , which range over the set of vertices on their respective half of the graph. Decoding the edge factor model  $h_{(i,j)}(y_i^e, y_j^f)$  simply involves finding the pair of tag assignments that gives the highest PMI score, subject to the dual constraints.

The way DD algorithms work in decomposing undirected graphical models is analogous to other message passing algorithms such as loopy belief propagation, but DD gives a stronger optimality guarantee upon convergence (Rush et al., 2010).

### 3 Joint Alignment and NER Decoding

In this section we develop an extended model in which NER information can in turn be used to improve alignment accuracy. Although we have seen more than a handful of recent papers that apply the dual decomposition method for joint inference problems, all of the past work deals with cases where the various model components have the same inference output space (e.g., dependency parsing (Koo et al., 2010), POS tagging (Rush et al., 2012), etc.). In our case the output space is the much more complex joint alignment and NER tagging space. We propose a novel dual decomposition variant for performing inference over this joint space.

Most commonly used alignment models, such as the IBM models and HMM-based aligner are unsupervised learners, and can only capture simple distortion features and lexical translational features due to the high complexity of the structure prediction space. On the other hand, the CRF-based NER models are trained on manually annotated data, and admit richer sequence and lexical features. The entity label predictions made by the NER model can potentially be leveraged to correct alignment mistakes. For example, in Figure 1, if the tagger knows that the word ‘‘Agency’’ is tagged I-ORG, and if it also knows that the first comma in the Chinese sentence is not part of any entity, then we can infer it is very unlikely that there exists an alignment link between ‘‘Agency’’ and the comma.

To capture this intuition, we extend the BI-NER model to jointly perform word alignment and NER decoding, and call the resulting model BI-NER-WA. As a first step, instead of taking the output from an aligner as fixed input, we incorporate two uni-directional aligners into our model. We name the Chinese-to-English aligner model as  $m(\mathbf{B}^e)$  and the reverse directional model  $n(\mathbf{B}^f)$ .  $\mathbf{B}^e$  is a matrix that holds the output of the Chinese-to-English aligner. Each  $b^e(i, j)$  binary variable in  $\mathbf{B}^e$  indicates whether  $f_j$  is aligned to  $e_i$ ; similarly we define output matrix  $\mathbf{B}^f$  and  $b^f(i, j)$  for Chinese. In our experiments, we used two HMM-based alignment models. But in principle we can adopt any alignment model as long as we can perform efficient inference over it.

We introduce a cross-lingual edge factor  $\zeta(i, j)$  in the undirected graphical model for every pair of word indices  $(i, j)$ , which predicts a binary vari-

**Algorithm 2** DD inference algorithm for joint alignment and NER model. A line marked with (2) means it applies to the BI-NER model; a line marked with (3) means it applies to the BI-NER-WA model.

---

```

 $S \leftarrow \mathcal{A}$  (2)
 $S \leftarrow \{(i, j) : \forall i \in |\mathbf{e}|, \forall j \in |\mathbf{f}|\}$  (3)
 $\forall i \in |\mathbf{e}| : w_i^e = 0; \forall j \in |\mathbf{f}| : w_j^f = 0$  (2,3)
 $\forall (i, j) \in \mathcal{S} : d^e(i, j) = 0, d^f(i, j) = 0$  (3)
for  $t \leftarrow 1$  to  $T$  do
   $\mathbf{y}^{e^{(k)*}} \leftarrow \operatorname{argmax}_k k(\mathbf{y}^{e^{(k)}}) + \sum_{i \in |\mathbf{e}|} w_i^e y_i^{e^{(k)}}$  (2,3)
   $\mathbf{y}^{f^{(l)*}} \leftarrow \operatorname{argmax}_l l(\mathbf{y}^{f^{(l)}}) + \sum_{j \in |\mathbf{f}|} w_j^f y_j^{f^{(l)}}$  (2,3)
   $\mathbf{B}^{e*} \leftarrow \operatorname{argmax}_m m(\mathbf{B}^e) + \sum_{(i,j)} d^e(i, j) b^e(i, j)$  (3)
   $\mathbf{B}^{f*} \leftarrow \operatorname{argmax}_n n(\mathbf{B}^f) + \sum_{(i,j)} d^f(i, j) b^f(i, j)$  (3)
  for all  $(i, j) \in \mathcal{S}$  do
     $(y_i^{e^{(h)*}}, y_j^{f^{(h)*}}) \leftarrow -w_i^e y_i^{e^{(h)}} - w_j^f y_j^{f^{(h)}} + \operatorname{argmax} h_{(i,j)}(y_i^{e^{(q)}} y_j^{f^{(q)}})$  (2)
     $(y_i^{e^{(q)*}}, y_j^{f^{(q)*}}) \leftarrow -w_i^e y_i^{e^{(q)}} - w_j^f y_j^{f^{(q)}} + \operatorname{argmax} q_{(i,j)}(y_i^{e^{(q)}} y_j^{f^{(q)}} a(i, j)) - d^e(i, j) a(i, j) - d^f(i, j) a(i, j)$  (3)
    end for
    Conv =  $(\mathbf{y}^{e^{(k)}} = \mathbf{y}^{e^{(q)}} \wedge \mathbf{y}^{f^{(l)}} = \mathbf{y}^{f^{(q)}})$  (2)
    Conv =  $(\mathbf{B}^e = \mathbf{A} = \mathbf{B}^f \wedge \mathbf{y}^{e^{(k)}} = \mathbf{y}^{e^{(q)}} \wedge \mathbf{y}^{f^{(l)}} = \mathbf{y}^{f^{(q)}})$  (3)
    if Conv = true, then
      return  $(\mathbf{y}^{e^{(k)*}}, \mathbf{y}^{f^{(l)*}})$  (2)
      return  $(\mathbf{y}^{e^{(k)*}}, \mathbf{y}^{f^{(l)*}}, \mathbf{A})$  (3)
    else
      for all  $i \in |\mathbf{e}|$  do
         $w_i^e \leftarrow w_i^e + \alpha_t (y_i^{e^{(q|h)*}} - y_i^{e^{(k)*}})$  (2,3)
      end for
      for all  $j \in |\mathbf{f}|$  do
         $w_j^f \leftarrow w_j^f + \alpha_t (y_j^{f^{(q|h)*}} - y_j^{f^{(l)*}})$  (2,3)
      end for
      for all  $(i, j) \in \mathcal{S}$  do
         $d^e(i, j) \leftarrow d^e(i, j) + \alpha_t (a^{e*}(i, j) - b^{e*}(i, j))$  (3)
         $d^f(i, j) \leftarrow d^f(i, j) + \alpha_t (a^{f*}(i, j) - b^{f*}(i, j))$  (3)
      end for
    end if
  end for
return  $(\mathbf{y}_{(\mathbf{T})}^{e^{(k)*}}, \mathbf{y}_{(\mathbf{T})}^{f^{(l)*}})$  (2)
return  $(\mathbf{y}_{(\mathbf{T})}^{e^{(k)*}}, \mathbf{y}_{(\mathbf{T})}^{f^{(l)*}}, \mathbf{A}_{(T)})$  (3)

```

---

able  $a(i, j)$  for an alignment link between  $e_i$  and  $f_j$ . The edge factor also predicts the entity tags for  $e_i$  and  $f_j$ .

The new edge potential  $q$  is defined as:

$$q_{(i,j)}(y_i^e, y_j^f, a(i, j)) = \log(P(a(i, j) = 1)) + S(y_i^e, y_j^f | a(i, j))^{P(a(i, j) = 1)}$$

$$S(y_i^e, y_j^f | a(i, j)) = \begin{cases} \text{pmi}(y_i^e, y_j^f), & \text{if } a(i, j) = 1 \\ 0, & \text{else} \end{cases}$$

$P(a(i, j) = 1)$  is the alignment probability assigned by the bilingual edge factor between node  $e_i$  and  $f_j$ . We initialize this value to  $\hat{P}(e_i, f_j) = \frac{1}{2}(P_m(e_i, f_j) + P_n(e_i, f_j))$ , where  $P_m(e_i, f_j)$  and  $P_n(e_i, f_j)$  are the posterior probabilities assigned by the HMM-aligners.

The joint optimization problem is defined as:

$$\max_{\mathbf{y}^{e^{(k)}} \mathbf{y}^{f^{(l)}} \mathbf{y}^{e^{(h)}} \mathbf{y}^{f^{(h)}} \mathbf{B}^e \mathbf{B}^f \mathbf{A}} k(\mathbf{y}^{e^{(k)}}) + l(\mathbf{y}^{f^{(l)}}) + m(\mathbf{B}^e) + n(\mathbf{B}^f) + \sum_{(i \in |\mathbf{e}|, j \in |\mathbf{f}|)} q_{(i,j)}(y_i^{e^h}, y_j^{f^h}, a(i, j))$$

$$\ni \forall (i, j) : (b^e(i, j) = a(i, j)) \wedge (b^f(i, j) = a(i, j))$$

$$\wedge \text{if } a(i, j) = 1 \text{ then } (y_i^{e^{(k)}} = y_i^{e^{(h)}}) \wedge (y_j^{f^{(l)}} = y_j^{f^{(h)}})$$

We include two dual constraints  $d^e(i, j)$  and  $d^f(i, j)$  over alignments for every bilingual edge factor  $\zeta(i, j)$ , which are applied to the English and Chinese sides of the alignment space, respectively.

The DD algorithm used for this model is given in Algorithm 2 (case 3). One special note is that after each iteration when we consider updates to the dual constraint for entity tags, we only check tag agreements for cross-lingual edge factors that have an alignment assignment value of 1. In other words, cross-lingual edges that are not aligned do not affect bilingual NER tagging.

Similar to  $\phi(i, j)$ ,  $\zeta(i, j)$  factors do not provide that much additional information other than some selectional preferences via PMI score. But the real power of these cross-language edge cliques is that they act as a liaison between the NER and alignment models on each language side, and encourage these models to indirectly agree with each other by having them all agree with the edge cliques.

It is also worth noting that since we decode the alignment models with Viterbi inference, additional constraints such as the neighborhood constraint proposed by DeNero and Macherey (2011) can be easily integrated into our model. The neighborhood constraint enforces that if  $f_j$  is aligned to  $e_i$ , then  $f_j$  can only be aligned to  $e_{i+1}$  or  $e_{i-1}$  (with a small penalty), but not any other word position. We report results of adding neighborhood constraints to our model in Section 6.

## 4 Experimental Setup

We evaluate on the large OntoNotes (v4.0) corpus (Hovy et al., 2006) which contains manually

annotated NER tags for both Chinese and English. Document pairs are sentence aligned using the Champollion Tool Kit (Ma, 2006). After discarding sentences with no aligned counterpart, a total of 402 documents and 8,249 parallel sentence pairs were used for evaluation. We will refer to this evaluation set as *full-set*. We use odd-numbered documents as the dev set and even-numbered documents as the blind test set. We did not perform parameter tuning on the dev set to optimize performance, instead we fix the initial learning rate to 0.5 and maximum iterations to 1,000 in all DD experiments. We only use the dev set for model development.

The Stanford CRF-based NER tagger was used as the monolingual component in our models (Finkel et al., 2005). It also serves as a state-of-the-art monolingual baseline for both English and Chinese. For English, we use the default tagger setting from Finkel et al. (2005). For Chinese, we use an improved set of features over the default tagger, which includes distributional similarity features trained on large amounts of non-overlapping data.<sup>1</sup>

We train the two CRF models on all portions of the OntoNotes corpus that are annotated with named entity tags, except the parallel-aligned portion which we reserve for development and test purposes. In total, there are about 660 training documents ( $\sim 16k$  sentences) for Chinese and 1,400 documents ( $\sim 39k$  sentences) for English.

Out of the 18 named entity types that are annotated in OntoNotes, which include person, location, date, money, and so on, we select the four most commonly seen named entity types for evaluation. They are *person*, *location*, *organization* and *GPE*. All entities of these four types are converted to the standard BIO format, and background tokens and all other entity types are marked with tag *O*. When we consider label agreements over aligned word pairs in all bilingual agreement models, we ignore the distinction between *B-* and *I-* tags.

We report standard NER measures (entity precision (P), recall (R) and  $F_1$  score) on the test set. Statistical significance tests are done using the paired bootstrap resampling method (Efron and Tibshirani, 1993).

For alignment experiments, we train two uni-

<sup>1</sup>The exact feature set and the CRF implementation can be found here: <http://nlp.stanford.edu/software/CRF-NER.shtml>

directional HMM models as our baseline and monolingual alignment models. The parameters of the HMM were initialized by IBM Model 1 using the agreement-based EM training algorithms from Liang et al. (2006). Each model is trained for 2 iterations over a parallel corpus of 12 million English words and Chinese words, almost twice as much data as used in previous work that yields state-of-the-art unsupervised alignment results (DeNero and Klein, 2008; Haghighi et al., 2009; DeNero and Macherey, 2011).

Word alignment evaluation is done over the sections of OntoNotes that have matching gold-standard word alignment annotations from GALE Y1Q4 dataset.<sup>2</sup> This subset contains 288 documents and 3,391 sentence pairs. We will refer to this subset as *wa-subset*. This evaluation set is over 20 times larger than the 150 sentences set used in most past evaluations (DeNero and Klein, 2008; Haghighi et al., 2009; DeNero and Macherey, 2011).

Alignments input to the B1-NER model are produced by thresholding the averaged posterior probability at 0.5. In joint NER and alignment experiments, instead of posterior thresholding, we take the direct intersection of the Viterbi-best alignment of the two directional models. We report the standard P, R,  $F_1$  and Alignment Error Rate (AER) measures for alignment experiments.

An important past work to make comparisons with is Burkett et al. (2010b). Their method is similar to ours in that they also model bilingual agreement in conjunction with two CRF-based monolingual models. But instead of using just the PMI scores of bilingual NE pairs, as in our work, they employed a feature-rich log-linear model to capture bilingual correlations. Parameters in their log-linear model require training with bilingually annotated data, which is not readily available. To counter this problem, they proposed an “up-training” method which simulates a supervised learning environment by pairing a weak classifier with strong classifiers, and train the bilingual model to rank the output of the strong classifier highly among the N-best outputs of the weak classifier. In order to compare directly with their method, we obtained the code behind Burkett et al. (2010b) and reproduced their experimental setting for the OntoNotes data. An extra set of 5,000 unannotated parallel sentence pairs are used for

<sup>2</sup>LDC Catalog No. LDC2006E86.

	Chinese			English		
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>
Mono	76.89	61.64	68.42	81.98	74.59	78.11
Burkett	77.52	65.84	71.20	82.28	76.64	79.36
Bi-soft	<b>79.14</b>	<b>71.55</b>	<b>75.15</b>	<b>82.58</b>	<b>77.96</b>	<b>80.20</b>

Table 1: NER results on bilingual parallel test set. Best numbers on each measure that are statistically significantly better than the monolingual baseline and Burkett et al. (2010b) are highlighted in bold.

training the reranker, and the reranker model selection was performed on the development dataset.

## 5 Bilingual NER Results

The main results on bilingual NER over the test portion of *full-set* are shown in Table 1. We initially experimented with the hard agreement model, but it performs quite poorly for reasons we discussed in Section 2.2. The BI-NER model with soft agreement constraints, however, significantly outperforms all baselines. In particular, it achieves an absolute F<sub>1</sub> improvement of 6.7% in Chinese and 2.1% in English over the CRF monolingual baselines.

A well-known issue with the DD method is that when the model does not necessarily converge, then the procedure could be very sensitive to hyper-parameters such as initial step size and early termination criteria. If a model only gives good performance with well-tuned hyper-parameters, then we must have manually annotated data for tuning, which would significantly reduce the applicability and portability of this method to other language pairs and tasks. To evaluate the parameter sensitivity of our model, we run the model from 50 to 3000 iterations before early stopping, and with 6 different initial step sizes from 0.01 to 1. The results are shown in Figure 2. The soft agreement model does not seem to be sensitive to initial step size and almost always converges to a superior solution than the baseline.

## 6 Joint NER and Alignment Results

We present results for the BI-NER-WA model in Table 2. By jointly decoding NER with word alignment, our model not only maintains significant improvements in NER performance, but also yields significant improvements to alignment performance. Overall, joint decoding with NER alone yields a 10.8% error reduction in AER over the baseline HMM-aligners, and also gives improve-

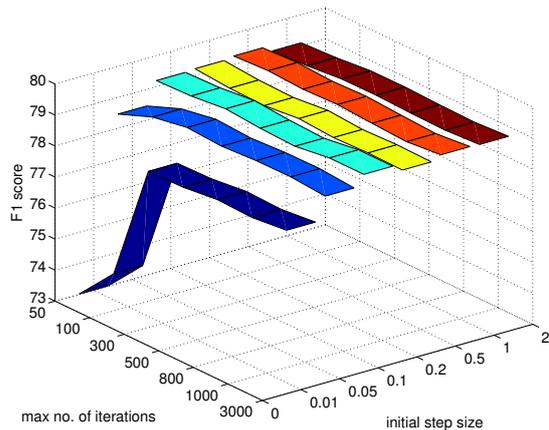


Figure 2: Performance variance of the soft agreement models on the Chinese dev dataset, as a function of step size (*x-axis*) and maximum number of iterations before early stopping (*y-axis*).

ment over BI-NER in NER. Adding additional neighborhood constraints gives a further 6% error reduction in AER, at the cost of a small loss in Chinese NER. In terms of word alignment results, we see great increases in F<sub>1</sub> and recall, but precision goes down significantly. This is because the joint decoding algorithm promotes an effect of “soft-union”, by encouraging the two unidirectional aligners to agree more often. Adding the neighborhood constraints further enhances this union effect.

## 7 Error Analysis and Discussion

We can examine the example in Figure 3 to gain an understanding of the model’s performance. In this example, a snippet of a longer sentence pair is shown with NER and word alignment results. The monolingual Chinese tagger provides a strong cue that word  $f_6$  is a person name because the unique 4-character word pattern is commonly associated with foreign names in Chinese, and also the word is immediately preceded by the word “president”. The English monolingual tagger, however, confuses the aligned word  $e_0$  with a GPE.

Our bilingual NER model is able to correct this error as expected. Similarly, the bilingual model corrects the error over  $e_{11}$ . However, the model also propagates labeling errors from the English side over the entity “Tibet Autonomous Region” to the Chinese side. Nevertheless, the resulting Chinese tags are arguably more useful than the original tags assigned by the baseline model.

In terms of word alignment, the HMM models failed badly on this example because of the long

	NER-Chinese			NER-English			word alignment			
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	AER
HMM-WA	-	-	-	-	-	-	<b>90.43</b>	40.95	56.38	43.62
Mono-CRF	82.50	66.58	73.69	84.24	78.70	81.38	-	-	-	-
Bi-NER	<b>84.87</b>	75.30	79.80	<b>84.47</b>	81.45	82.93	-	-	-	-
Bi-NER-WA	84.42	<b>76.34</b>	<b>80.18</b>	84.25	<b>82.20</b>	<b>83.21</b>	77.45	50.43	61.09	38.91
Bi-NER-WA+NC	84.25	75.09	79.41	84.28	82.17	<b>83.21</b>	76.67	<b>54.44</b>	<b>63.67</b>	<b>36.33</b>

Table 2: Joint alignment and NER test results. +NC means incorporating additional neighbor constraints from DeNero and Macherey (2011) to the model. Best number in each column is highlighted in bold.

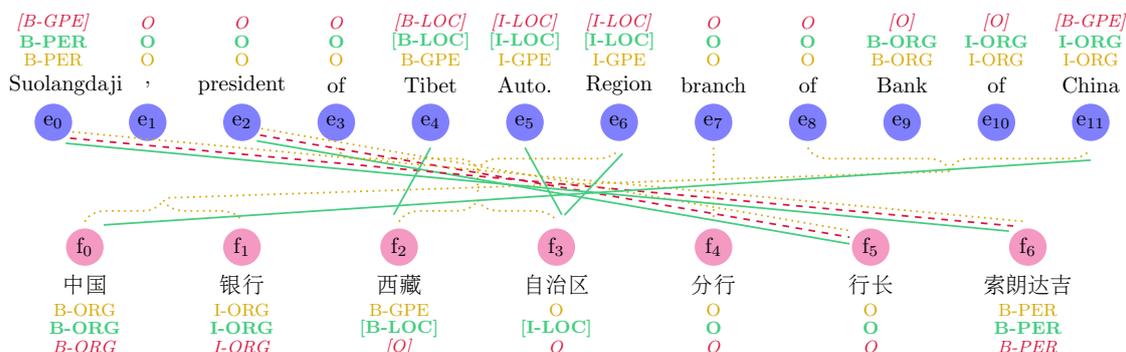


Figure 3: An example output of our BI-NER-WA model. Dotted alignment links are the oracle, dashed links are alignments from HMM baseline, and solid links are outputs of our model. Entity tags in the gold line (closest to nodes  $e_i$  and  $f_j$ ) are the gold-standard tags; in the green line (second closest to nodes) are output from our model; and in the crimson line (furthest from nodes) are baseline output.

distance swapping phenomena. The two unidirectional HMMs also have strong disagreements over the alignments, and the resulting baseline aligner output only recovers two links. If we were to take this alignment as fixed input, most likely we would not be able to recover the error over  $e_{11}$ , but the joint decoding method successfully recovered 4 more links, and indirectly resulted in the NER tagging improvement discussed above.

## 8 Related Work

The idea of employing bilingual resources to improve over monolingual systems has been explored by much previous work. For example, Huang et al. (2009) improved parsing performance using a bilingual parallel corpus. In the NER domain, Li et al. (2012) presented a cyclic CRF model very similar to our BI-NER model, and performed approximate inference using loopy belief propagation. The feature-rich CRF formulation of bilingual edge potentials in their model is much more powerful than our simple PMI-based bilingual edge model. Adding a richer bilingual edge model might well further improve our results, and this is a possible direction for further experimentation. However, a big drawback of this ap-

proach is that training such a feature-rich model requires manually annotated bilingual NER data, which can be prohibitively expensive to generate. How and where to obtain training signals without manual supervision is an interesting and open question. One of the most interesting papers in this regard is Burkett et al. (2010b), which explored an “up-training” mechanism by using the outputs from a strong monolingual model as ground-truth, and simulated a learning environment where a bilingual model is trained to help a “weakened” monolingual model to recover the results of the strong model. It is worth mentioning that since our method does not require additional training and can take pretty much any existing model as “black-box” during decoding, the richer and more accurate bilingual model learned from Burkett et al. (2010b) can be directly plugged into our model.

A similar dual decomposition algorithm to ours was proposed by Riedel and McCallum (2011) for biomedical event detection. In their Model 3, the trigger and argument extraction models are reminiscent of the two monolingual CRFs in our model; additional binding agreements are enforced over every protein pair, similar to how we enforce agreement between every aligned word

pair. Martins et al. (2011b) presented a new DD method that combines the power of DD with the augmented Lagrangian method. They showed that their method can achieve faster convergence than traditional sub-gradient methods in models with many overlapping components (Martins et al., 2011a). This method is directly applicable to our work.

Another promising direction for improving NER performance is in enforcing global label consistency across documents, which is an idea that has been greatly explored in the past (Sutton and McCallum, 2004; Bunescu and Mooney, 2004; Finkel et al., 2005). More recently, Rush et al. (2012) and Chieu and Teow (2012) have shown that combining local prediction models with global consistency models, and enforcing agreement via DD is very effective. It is straightforward to incorporate an additional global consistency model into our model for further improvements.

Our joint alignment and NER decoding approach is inspired by prior work on improving alignment quality through encouraging agreement between bi-directional models (Liang et al., 2006; DeNero and Macherey, 2011). Instead of enforcing agreement in the alignment space based on best sequences found by Viterbi, we could opt to encourage agreement between posterior probability distributions, which is related to the posterior regularization work by Graça et al. (2008). Cromières and Kurohashi (2009) proposed an approach that takes phrasal bracketing constraints from parsing outputs, and uses them to enforce phrasal alignments. This idea is similar to our joint alignment and NER approach, but in our case the phrasal constraints are indirectly imposed by entity spans. We also differ in the implementation details, where in their case belief propagation is used in both training and Viterbi inference.

Burkett et al. (2010a) presented a supervised learning method for performing joint parsing and word alignment using log-linear models over parse trees and an ITG model over alignment. The model demonstrates performance improvements in both parsing and alignment, but shares the common limitations of other supervised work in that it requires manually annotated bilingual joint parsing and word alignment data.

Chen et al. (2010) also tackled the problem of joint alignment and NER. Their method employs a

set of heuristic rules to expand a candidate named entity set generated by monolingual taggers, and then rank those candidates using a bilingual named entity dictionary. Our approach differs in that we provide a probabilistic formulation of the problem and do not require pre-existing NE dictionaries.

## 9 Conclusion

We introduced a graphical model that combines two HMM word aligners and two CRF NER taggers into a joint model, and presented a dual decomposition inference method for performing efficient decoding over this model. Results from NER and word alignment experiments suggest that our method gives significant improvements in both NER and word alignment. Our techniques make minimal assumptions about the underlying monolingual components, and can be adapted for many other tasks such as parsing.

## Acknowledgments

The authors would like to thank Rob Voigt and the three anonymous reviewers for their valuable comments and suggestions. We gratefully acknowledge the support of the National Natural Science Foundation of China (NSFC) via grant 61133012, the National “863” Project via grant 2011AA01A207 and 2012AA011102, the Ministry of Education Research of Social Sciences Youth funded projects via grant 12YJCZH304, and the support of the U.S. Defense Advanced Research Projects Agency (DARPA) Broad Operational Language Translation (BOLT) program through IBM.

Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, or the US government.

## References

- Yaser Al-Onaizan and Kevin Knight. 2002. Translating named entities using monolingual and bilingual resources. In *Proceedings of ACL*.
- Bogdan Babych and Anthony Hartley. 2003. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT*.

- Dimitri P. Bertsekas. 1999. *Nonlinear Programming*. Athena Scientific, New York.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1991. Word-sense disambiguation using statistical methods. In *Proceedings of ACL*.
- Razvan Bunescu and Raymond J. Mooney. 2004. Collective information extraction with relational Markov networks. In *Proceedings of ACL*.
- David Burkett, John Blitzer, and Dan Klein. 2010a. Joint parsing and alignment with weakly synchronized grammars. In *Proceedings of NAACL-HLT*.
- David Burkett, Slav Petrov, John Blitzer, and Dan Klein. 2010b. Learning better monolingual models with unannotated bilingual text. In *Proceedings of CoNLL*.
- Yufeng Chen, Chengqing Zong, and Keh-Yih Su. 2010. On jointly recognizing and aligning bilingual named entities. In *Proceedings of ACL*.
- Hai Leong Chieu and Loo-Nin Teow. 2012. Combining local and non-local information with dual decomposition for named entity recognition from text. In *Proceedings of 15th International Conference on Information Fusion (FUSION)*.
- Fabien Cromières and Sadao Kurohashi. 2009. An alignment algorithm using belief propagation and a structure-based distortion model. In *Proceedings of EACL/IJCNLP*.
- John DeNero and Dan Klein. 2008. The complexity of phrase alignment problems. In *Proceedings of ACL*.
- John DeNero and Klaus Macherey. 2011. Model-based aligner combination using dual decomposition. In *Proceedings of ACL*.
- Brad Efron and Robert Tibshirani. 1993. *An Introduction to the Bootstrap*. Chapman & Hall, New York.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of ACL*.
- Joao Graça, Kuzman Ganchev, and Ben Taskar. 2008. Expectation maximization and posterior constraints. In *Proceedings of NIPS*.
- Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised ITG models. In *Proceedings of ACL*.
- Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: the 90% solution. In *Proceedings of NAACL-HLT*.
- Fei Huang and Stephan Vogel. 2002. Improved named entity translation and bilingual named entity extraction. In *Proceedings of the 2002 International Conference on Multimodal Interfaces (ICMI)*.
- Liang Huang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of EMNLP*.
- Sungchul Kim, Kristina Toutanova, and Hwanjo Yu. 2012. Multilingual named entity recognition using parallel data and metadata from Wikipedia. In *Proceedings of ACL*.
- Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of EMNLP*.
- Qi Li, Haibo Li, Heng Ji, Wen Wang, Jing Zheng, and Fei Huang. 2012. Joint bilingual name tagging for parallel corpora. In *Proceedings of CIKM*.
- Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of HLT-NAACL*.
- Xiaoyi Ma. 2006. Champollion: A robust parallel text sentence aligner. In *Proceedings of LREC*.
- André F. T. Martins, Noah A. Smith, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2011a. Dual decomposition with many overlapping components. In *Proceedings of EMNLP*.
- Andre F. T. Martins, Noah A. Smith, Eric P. Xing, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2011b. Augmenting dual decomposition for map inference. In *Proceedings of the International Workshop on Optimization for Machine Learning (OPT 2010)*.
- Sebastian Riedel and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of EMNLP*.
- Alexander M. Rush and Michael Collins. 2012. A tutorial on dual decomposition and Lagrangian relaxation for inference in natural language processing. *JAIR*, 45:305–362.
- Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of EMNLP*.
- Alexander M. Rush, Roi Reichert, Michael Collins, and Amir Globerson. 2012. Improved parsing and POS tagging using inter-sentence consistency constraints. In *Proceedings of EMNLP*.
- Charles Sutton and Andrew McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *Proceedings of ICML Workshop on Statistical Relational Learning and Its connections to Other Fields*.

# Resolving Entity Morphs in Censored Data

Hongzhao Huang<sup>1</sup>, Zhen Wen<sup>2</sup>, Dian Yu<sup>1</sup>, Heng Ji<sup>1</sup>,  
Yizhou Sun<sup>3</sup>, Jiawei Han<sup>4</sup>, He Li<sup>5</sup>

<sup>1</sup>Computer Science Department and Linguistics Department,

Queens College and Graduate Center, City University of New York, New York, NY, USA

<sup>2</sup>IBM T. J. Watson Research Center, Hawthorne, NY, USA

<sup>3</sup>College of Computer and Information Science, Northeastern University, Boston, MA, USA

<sup>4</sup>Computer Science Department, University of Illinois at Urbana-Champaign, Urbana, IL, USA

<sup>5</sup>Admaster Inc., China

{hongzhaohuang<sup>1</sup>, yudiandoris<sup>1</sup>, hengjicuny<sup>1</sup>, liheact<sup>5</sup>}@gmail.com,

zhenwen@us.ibm.com<sup>2</sup>, yzsun@ccs.neu.edu<sup>3</sup>, hanj@illinois.edu<sup>4</sup>

## Abstract

In some societies, internet users have to create information morphs (e.g. “Peace West King” to refer to “Bo Xilai”) to avoid active censorship or achieve other communication goals. In this paper we aim to solve a new problem of resolving entity morphs to their real targets. We exploit temporal constraints to collect cross-source comparable corpora relevant to any given morph query and identify target candidates. Then we propose various novel similarity measurements including surface features, meta-path based semantic features and social correlation features and combine them in a learning-to-rank framework. Experimental results on Chinese Sina Weibo data demonstrate that our approach is promising and significantly outperforms baseline methods<sup>1</sup>.

## 1 Introduction

Language constantly evolves to maximize communicative success and expressive power in daily social interactions. The proliferation of online social media significantly expedites this evolution, as new phrases triggered by social events may be disseminated rapidly in social media. To automatically analyze such fast evolving language in social media, new computational models are demanded.

In this paper, we focus on one particular language evolution that creates new ways to communicate sensitive subjects because of the existence of internet information censorship. We call this

<sup>1</sup>Some of the resources and open source programs developed in this work are made freely available for research purpose at <http://nlp.cs.qc.cuny.edu/Morphing.tar.gz>

phenomenon *information morph*. For example, when Chinese online users talk about the former politician “Bo Xilai”, they use a morph “Peace West King” instead, a historical figure four hundreds years ago who governed the same region as Bo. Morph can be considered as a special case of alias used for hiding true entities in malicious environment (Hsiung et al., 2005; Pantel, 2006). However, social network plays an important role in generating morphs. Usually morphs are generated by harvesting the collective wisdom of the crowd to achieve certain communication goals. Aside from the purpose of avoiding censorship, other motivations for using morph include expressing sarcasm/irony, positive/negative sentiment or making descriptions more vivid toward some entities or events. Table 1 presents the wide range of cases that are used to create the morphs. We can see that a morph can be either a regular term with new meaning or a newly created term.

Morph	Target	Motivation
Peace West King	Bo Xilai	Sensitive
Blind Man	Chen Guangcheng	Sensitive
Miracle Brother	Wang Yongping	Irony
Kim Fat	Kim Joing-il	Negative
Kimchi Country	South Korea	Vivid

Table 1: Morph Examples and Motivations.

We believe that successful resolution of morphs is a crucial step for automated understanding of the fast evolving social media language, which is important for social media marketing (Barwise and Meehan, 2010). Another application is to help common users without enough background/cultural knowledge to understand internet language for their daily use. Furthermore, our approaches can also be applied for satire or other implicit meaning recognition, as well as information extraction (Bollegala et al., 2011).

However, morph resolution in social media is challenging due to the following reasons. First, the sensitive real targets that exist in the same data source under active censorship are often automatically filtered. Table 2 presents the distributions of some examples of morphs and their targets in English Twitter and Chinese Sina Weibo. For example, the target “*Chen Guangcheng*” only appears once in Weibo. Thus, the co-occurrence of a morph and its target is quite low in the vast amount of information in social media. Second, most morphs were not created based on pronunciations, spellings or other encryptions of their original targets. Instead, they were created according to semantically related entities in historical and cultural narratives (e.g. “*Peace West King*” as morph of “*Bo Xilai*”) and thus very difficult to capture based on typical lexical features. Third, tweets from Twitter/Chinese Weibo are short (only up to 140 characters) and noisy, resulting in difficult extraction of rich and accurate evidences due to the lack of enough contexts.

Morph	Target	Frequency in Twitter		Frequency in Weibo	
		Morph	Target	Morph	Target
Hu Ji	Hu Jintao	1	3,864	2,611	71
Blind Man	Chen Guangcheng	18	2,743	20,941	1
Baby	Wen Jiabao	2238	2021	26,279	8

Table 2: Distributions of Morph Examples

To the best of our knowledge, this is the first work to use NLP and social network analysis techniques to automatically resolve morphed information. To address the above challenges, our paper offers the following novel contributions.

- We detect target candidates by exploiting the dynamics of the social media to extract temporal distribution of entities, based on the assumption that the popularity of an individual is correlated between censored and uncensored text within a certain time window.
- Our approach builds and analyzes heterogeneous information networks from multiple sources, such as Twitter, Sina Weibo and web documents in formal genre (e.g. news) because a morph and its target tend to appear in similar contexts.
- We propose two new similarity measures, as well as integrating temporal information into

the similarity measures to generate global semantic features.

- We model social user behaviors and use social correlation to assist in measuring semantic similarities because the users who posted a morph and its corresponding target tend to share similar interests and opinions.

Our experiments demonstrate that the proposed approach significantly outperforms traditional alias detection methods (Hsiung et al., 2005).

## 2 Approach Overview

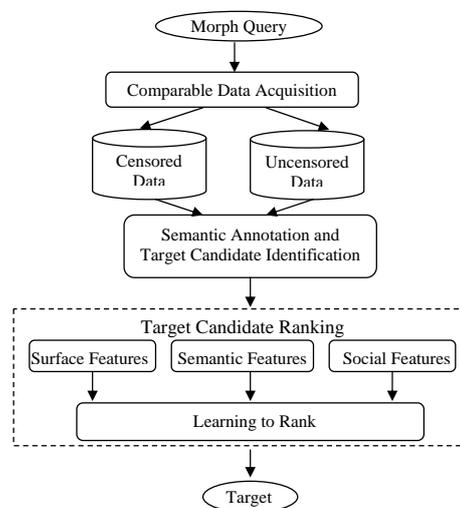


Figure 1: Overview of Morph Resolution

Given a morph query  $m$ , the goal of morph resolution is to find its real target. Figure 1 depicts the general procedure of our approach. It consists of two main sub-tasks:

- **Target Candidate Identification:** For each  $m$ , discover a list of target candidates  $E = \{e_1, e_2, \dots, e_N\}$ . First, relevant comparable data sets that include  $m$  are retrieved. In this paper we collect comparable censored data from Weibo and uncensored data from Twitter and Web documents such as news articles. We then apply various annotations such as word segmentation, part-of-speech tagging, noun phrase chunking, name tagging and event extraction to these data sets.
- **Target Candidate Ranking:** Rank the target candidates in  $E$ . We explore various features including surface, semantic and social features, and incorporate them into a learning to

rank framework. Finally, the top ranked candidate is produced as the resolved target.

### 3 Target Candidate Identification

The general goal of the first step is to identify a list of target candidates for each morph query from the comparable corpora including Sina Weibo, Chinese News websites and English Twitter. However, obviously we cannot consider all of the named entities in these sources as target candidates due to the sheer volume of information. In addition, morphs are not limited to named entity forms. In order to narrow down the scope of target candidates, we propose a *Temporal Distribution Assumption* as follows. The intuition is that a morph  $m$  and its real target  $e$  should have similar temporal distributions in terms of their occurrences. Suppose the data sets are separated into  $Z$  temporal slots (e.g. by day), the assumption can be stated as:

Let  $T_m = \{t_{m1}, t_{m2}, \dots, t_{mZ_m}\}$  be the set of temporal slots each morph  $m$  occurs, and  $T_e = \{t_{e1}, t_{e2}, \dots, t_{eZ_e}\}$  be the set of slots a target candidate  $e$  occurs. Then  $e$  is considered as a target candidate of  $m$  if and only if, for each  $t_{mi} \in T_m$  ( $i = 1, 2, \dots, Z_m$ ), there exist a  $j \in \{1, 2, \dots, Z_e\}$  such that  $t_{mi} - t_{ej} \leq \delta$ , where  $\delta$  is a threshold value (in this paper we set the threshold to 7 days, which is optimized from a development set). For comparison we also attempted topic modeling approach to detect target candidates, as shown in section 5.3.

### 4 Target Candidate Ranking

Next, we propose a learning-to-rank framework to rank target candidates based on various levels of novel features based on surface, semantic and social analysis.

#### 4.1 Surface Features

We first extract surface features between the morph and the candidate based on measuring orthographic similarity measures which were commonly used in entity coreference resolution (e.g. (Ng, 2010; Hsiung et al., 2005)). The measures we use include “string edit distance”, “normalized string edit distance” (Wagner and Fischer, 1974) and “longest common subsequence” (Hirschberg, 1977).

## 4.2 Semantic Features

### 4.2.1 Motivations

Fortunately, although a morph and its target may have very different orthographic forms, they tend to be embedded in *similar semantic contexts* which involve similar topics and events. Figure 2 presents some example messages under censorship (Weibo) and not under censorship (Twitter and Chinese Daily). We can see that they include similar topics, events (e.g., “fell from power”, “gang crackdown”, “sing red songs”), and semantic relations (e.g., family relations with “Bo Guagua”). Therefore if we can automatically extract and exploit these indicative semantic contexts, we can narrow down the real targets effectively.



Figure 2: Cross-source Comparable Data Example (each morph and target pair is shown in the same color)

### 4.2.2 Information Network Construction

We define an information network as a directed graph  $G = (\mathcal{V}, \mathcal{E})$  with an object type mapping function  $\tau : \mathcal{V} \rightarrow \mathcal{A}$  and a link type mapping function  $\phi : \mathcal{E} \rightarrow \mathcal{R}$ , where each object  $v \in \mathcal{V}$  belongs to one particular object type  $\tau(v) \in \mathcal{A}$ , each link  $e \in \mathcal{E}$  belongs to a particular relation  $\phi(e) \in \mathcal{R}$ . If two links belong to the same relation type, then they share the same starting object type as well as the same ending object type. An information network is *homogeneous* if and only if there is only one type for both objects and links, and an information network is *heterogeneous* when the objects are from multiple distinct types or there exist more than one type of links.

In order to construct the information networks for morphs, we apply the Stanford Chinese word

segmenter with Chinese Penn Treebank segmentation standard (Chang et al., 2008) and Stanford part-of-speech tagger (Toutanova et al., 2003) to process each sentence in the comparable data sets. Then we apply a hierarchical Hidden Markov Model (HMM) based Chinese lexical analyzer ICTCLAS (Zhang et al., 2003) to extract named entities, noun phrases and events.

We have also attempted using the results from Dependency Parsing, Relation Extraction and Event Extraction tools (Ji and Grishman, 2008) to enrich the link types. Unfortunately the state-of-the-art techniques for these tasks still perform poorly on social media in terms of both accuracy and coverage of important information, these sophisticated semantic links all produced negative impact on the target ranking performance. Therefore we limited the types of vertices into: *Morph* (M), *Entity*(E), which includes target candidates, *Event* (EV), and *Non-Entity Noun Phrases* (NP); and used *co-occurrence* as the edge type. We extract entities, events, and non-entity noun phrases that occur in more than one tweet as neighbors. And for two vertices  $x_i$  and  $x_j$ , the weight  $w_{ij}$  of their edge is the frequency they co-occur together within the tweets. A network schema of such networks is shown in Figure 3. Figure 4

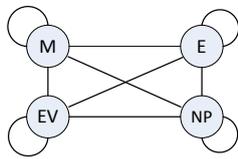


Figure 3: Network Schema of Morph-Related Heterogeneous Information Network

presents an example of a heterogeneous information network from the motivation examples following the above network schema, which connects the morphs “*Peace West King*”, “*Buhou*” and their corresponding target “*Bo Xilai*”.

#### 4.2.3 Meta-Path-Based Semantic Similarity Measurements

Given the constructed network, a straightforward solution for finding the target for a morph is to use link-based similarity search. However, now objects are linked to different types of neighbors, if all neighbors are treated as the same, it may cause information loss problems. For example, the entity “*重庆 (Chongqing)*” is a very important aspect characterizing the politician “*薄熙来 (Bo Xilai)*”

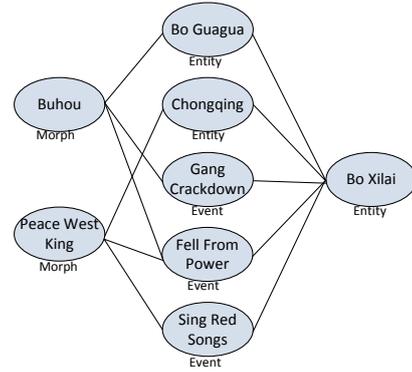


Figure 4: Example of Morph-Related Heterogeneous Information Network

since he governed it, and if a morph  $m$  which is also highly correlated with “*重庆 (Chongqing)*”, it is very likely that “*Bo Xilai*” is the real target of  $m$ . Therefore, the semantic features generated from neighbors such as the entity “*重庆 (Chongqing)*” should be treated differently from other types of neighbors such as “*人才 (talented people)*”.

In this work, we propose to measure the similarity of two nodes over heterogeneous networks as shown in Figure 3, by distinguishing neighbors into three types according to the network schema (i.e. entities, events, non-entity noun phrases). We then adopt meta-path-based similarity measures (Sun et al., 2011a; Sun et al., 2011b), which are defined over heterogeneous networks to extract semantic features. A meta-path is a path defined over a network, and composed of a sequence of relations between different object types. For example, as shown in Figure 3, a morph and its target candidate can be connected by three meta-paths, including “*M - E - E*”, “*M - EV - E*”, and “*M - NP - E*”. Intuitively, each meta-path provides a unique angle to measure how similar two objects are.

For the determined meta-paths, we extract semantic features using the similarity measures proposed in (Sun et al., 2011a; Hsiung et al., 2005). We denote the neighbor sets of certain type for a morph  $m$  and a target candidate  $e$  as  $\Gamma(m)$  and  $\Gamma(e)$ , and a meta-path as  $\mathcal{P}$ . We now list several meta-path-based similarity measures below.

*Common neighbors (CN)*. It measures the number of common neighbors that  $m$  and  $e$  share as  $|\Gamma(m) \cap \Gamma(e)|$ .

*Path count (PC)*. It measures the number of path instances between  $m$  and  $e$  following meta-path  $\mathcal{P}$ .

*Pairwise random walk (PRW)*. For a meta-path  $\mathcal{P}$  that can be decomposed into two shorter

meta-paths with the same length  $\mathcal{P} = (\mathcal{P}_1\mathcal{P}_2)$ , pairwise random walk measures the probability of the pairwise random walk starting from both  $m$  and  $e$  and reaching the same middle object. More formally, it is computed as  $\sum_{(p_1 p_2) \in (\mathcal{P}_1 \mathcal{P}_2)} prob(p_1)prob(p_2^{-1})$ , where  $p_2^{-1}$  is the inverse of  $p_2$ .

**Kullback-Leibler distance (KLD).** For  $m$  and  $e$ , the pairwise random walk probability of their neighbors can be represented as two probability vectors, then Kullback-Leibler distance (Hsiung et al., 2005) can be used to compute  $sim(m, e)$ .

Beyond the above similarity measures, we also propose to use cosine-similarity-style normalization method to modify common neighbor and pairwise random walk measures so that we can ensure the morph node and the target candidate node are strongly connected and also have similar popularity. The modified algorithms penalize features involved with the highly popular objects, since they are more likely to have accidental interactions with each other.

**Normalized common neighbors (NCN).** Normalized common neighbors can be measured as  $sim(m, e) = \frac{|\Gamma(m) \cap \Gamma(e)|}{\sqrt{|\Gamma(m)|} \sqrt{|\Gamma(e)|}}$ . It refines the simple counting of common neighbors by avoiding bias to highly visible or concentrated objects.

**Pairwise random walk/cosine (PRW/cosine).** Pairwise random walk measures linkage weights disproportionately with their visibility to their neighbors, which may be too strong. Instead, we propose to use a tamer normalization method as  $\sum_{(p_1 p_2) \in (\mathcal{P}_1 \mathcal{P}_2)} f(p_1)f(p_2^{-1})$ , where.

$$f(p_1) = \frac{count(m, x)}{\sqrt{\sum_{x \in \Omega} count(m, x)}},$$

$$f(p_2) = \frac{count(e, x)}{\sqrt{\sum_{x \in \Omega} count(e, x)}},$$

and  $\Omega$  is the set of middle objects connecting the decomposed meta-paths  $p_1$  and  $p_2^{-1}$ ,  $count(y, x)$  is the total number of paths between  $y$  and the middle object  $x$ ,  $y$  could be  $m$  or  $e$ .

The above similarity measures can also be applied to homogeneous networks that do not differentiate the neighbor types.

#### 4.2.4 Global Semantic Feature Generation

A morph tends to have higher temporal correlation with its real target, and share more similar topics compared to other irrelevant targets. Therefore, we propose to incorporate temporal information

into similarity measures to generate global semantic features.

Let  $T = t_1 \cup t_2 \cup \dots \cup t_N$  be a set of temporal slots (i.e. by day),  $E$  be the set of target candidates for each morph  $m$ . Then for each  $t_i \in T$ , and each  $e \in E$ , the local semantic features  $sim_{t_i}(m, e)$  is extracted based only on the information posted within  $t_i$  using one of the similarity measures introduced in Section 4.2.3. Then we propose two approaches to generate global semantic features. The first approach is adding the similarity score between  $m$  and  $e$  in each temporal slot to attain the first set of global features:

$$sim_{global\_sum}(m, e) = \sum_{t_i \in T} sim_{t_i}(m, e).$$

The second method first normalizes the similarity score in each temporal slot  $t_i$ , then sum the normalized scores to generate the second set of global features, which can be calculated as

$$sim_{global\_norm}(m, e) = \sum_{t_i \in T} norm_{t_i}(m, e).$$

where  $norm_{t_i}(m, e) = \frac{sim_{t_i}(m, e)}{\sum_{e \in E} sim_{t_i}(m, e)}$ .

#### 4.2.5 Integrate Cross Source/Cross Genre Information

Due to internet information censorship or surveillance, users may need to use morphs to post sensitive information. For example, the Chinese Weibo message “都进去了,还要贡着不厚吗 (*Already put in prison, still need to serve Buhou?*)” include a morph 不厚 (*Buhou*). In contrast, users are less restricted in some other uncensored social media such as Twitter. For example, the tweet from Twitter “...把薄熙来称作“平西王”或者“不厚”... (...call Bo Xilai “peace west king” or “buhou”...)” contains both the morph and the real target 薄熙来 (*Bo Xilai*). Therefore, we propose to integrate information from another source (e.g. Twitter) to help resolution of sensitive morphs in Weibo.

Another difficulty from morph resolution in micro-blogging is that tweets are only allowed to contain maximum 140 characters with a lot of noise and diverse topics. The shortness and diversity of tweets may limit the power of content analysis for semantic feature extraction. However, formal genres such as web documents are cleaner and contain richer contexts, thus can provide more topically related information. In this work, we also exploit the background web documents from the

embedded URLs in tweets to enrich information network construction. After applying the same annotation techniques as tweets for uncensored data sets, sentence-level co-occurrence relations are extracted and integrated into the network as shown in Figure 3.

### 4.3 Social Features

It has been shown that there exist correlation between neighbors in social networks (Anagnostopoulos et al., 2008; Wen and Lin, 2010). Because of such social correlation, close social neighbors in social media such as Twitter and Weibo may post similar information, or share similar opinion. Therefore, we can utilize social correlation to assist in resolving morphs.

As social correlation can be defined as a function of social distance between a pair of users, we use social distance as a proxy to social correlation in our approach. The social distance between user  $i$  and  $j$  is defined by considering the degree of separation in their interaction (e.g. retweeting and mentioning) and the amount of the interaction. Similar definition has been shown effective in characterizing social distance in social networks extracted from communication data (Lin et al., 2012; Wen and Lin, 2010). Specifically, it is  $dist(i, j) = \sum_{k=1}^{K-1} \frac{1}{strength(v_k, v_{k+1})}$ , where  $v_1, \dots, v_k$  are the nodes on the shortest path from user  $i$  to user  $j$ , and  $strength(v_k, v_{k+1})$  measures the strength of interactions between  $v_k$  and  $v_{k+1}$  as:  $strength(i, j) = \frac{\log(X_{ij})}{\max_j \log(X_{ij})}$ , where  $X_{ij}$  is the total interactions between user  $i$  and  $j$ , including both retweeting and mentioning (If  $X_{ij} < 10$ , we set  $strength(i, j) = 0$ ).

We integrate social correlation and temporal information to define our social features. The intuition is that when a morph is used by an user, the real target may also in the posts by the user or his/her close friends within a certain time period. Let  $T$  be the set of temporal slots a morph  $m$  occurs,  $U_t$  be the set of users whose posts include  $m$  in slot  $t$  where  $t \in T$ , and  $U_c$  be the set of close friends (i.e., social distance  $< 0.5$ ) for  $U_t$ . The social features are defined as

$$s(m, e) = \frac{\sum_{t \in T} f(e, t, U_t, U_c)}{|T|}.$$

where  $f(e, t, U_t, U_c)$  is a indicator function which return 1 if one of the users in  $U_t$  or  $U_c$  posts tweets include the target candidate  $e$  within 7 days before  $t$ .

### 4.4 Learning-to-Rank

Similar to (Hsiung et al., 2005; Sun et al., 2011a), we then model the probability of linkage prediction between a morph  $m$  and its target candidate  $e$  as a function incorporating the surface, semantic and social features. Given a training pair  $\langle m, e \rangle$ , we choose the standard logistic regression model to learn weights for the features defined above. The learnt model is used to predict the probability of linking an unseen morph and its target candidate. Based on the descending ranking order of the probability, we select top  $k$  candidates as the final answers based on the answer size  $k$ .

## 5 Experiments

Next, we present the experiment under various settings shown in Table 3, and the impacts of cross source and cross genre information.

### 5.1 Data and Evaluation Metric

We collected 1,553,347 tweets from Chinese Sina Weibo from May 1 to June 30 to construct the censored data set, and retrieved 66,559 web documents from the embedded URLs in tweets as the initial uncensored data set. Retweets and redundant web documents are filtered to ensure more reliable frequency counting of co-occurrence relations. We asked two native Chinese annotators to analyze the data, and construct a test set consisted of 107 morph entities (81 persons and 26 locations) and their real targets as our references. We verified the references by Web resources including the summary of popular morphs in Wikipedia<sup>2</sup>. In addition, we used 23 sensitive morphs and the entities that appear in the tweets as queries and retrieved 25,128 Chinese tweets from 10% Twitter feeds within the same time period, as well as 7,473 web documents from the embedded URLs and added them into the uncensored data set.

To evaluate the system performance, we use leave-one-out cross validation by computing accuracy as  $Acc@k = \frac{C_k}{Q}$ , where  $C_k$  is the total number of correctly resolved morphs at top  $k$  ranked answers, and  $Q$  is the total number of morph queries. We consider a morph as correctly resolved at the top  $k$  answers if the top  $k$  answer set contains the real target of the morph.

<sup>2</sup><http://zh.wikipedia.org/wiki/中国大陆网络语言列表>

Feature sets	Descriptions
<b>Surf</b>	Surface features
<b>HomB</b>	Semantic features extracted from homogeneous CN, PC, PRW, and KLD
<b>HomE</b>	HomB + semantic features extracted from homogeneous NCN and PRW/cosine
<b>HetB</b>	Semantic features extracted from heterogeneous CN, PC, PRW and KLD
<b>HetE</b>	HetB + Semantic features extracted from heterogeneous NCN and PRW/cosine
<b>Glob*</b>	Global semantic features
<b>Social</b>	Social network features

Table 3: Description of feature sets. \* Glob only uses the same set of similarity measures when combined with other semantic features.

## 5.2 Resolution Performance

### 5.2.1 Single Genre Information

We first study the contributions of each set of surface and semantic features, as shown in the first five rows in Table 4. The poor performance based on surface features shows that morph resolution task is very challenging since 70% of morphs are not orthographically similar to their real targets. Thus, capturing a morph’s semantic meaning is crucial. Overall, the results demonstrate the effectiveness of our proposed methods. Specifically, comparing “HomB” and “HetB”, “HomE” and “HetE”, we can see that the semantic features based on heterogeneous networks have advantages over those based on homogeneous networks. This corroborates that different neighbor sets contribute differently, and such discrepancies should be captured. And comparisons of “HomB” and “HomE”, “HetB” and “HetE” demonstrate the effectiveness of our two new proposed measures. To evaluate the importance of each similarity measures, we delete the semantic features obtained from each measure in “HetE” and re-evaluate the system. We find that NCN is the most effective measure, while KLD is the least important one. Further adding the global semantic features significantly improves the performance. This indicates that capturing both temporal correlations and semantics of morphing simultaneously are important for morph resolution.

Table 5 shows that combination of surface and semantic features further improves the performance, showing that they are complementary. For example, using only surface features, the real target “乔布斯 (Steve Jobs)” of the morph “乔帮主 (Qiao Boss)” is not top ranked since some other candidates such as “乔治 (George)” are more orthographically similar. However, “Steve Jobs” is ranked top when combined with semantic features.

Features	Surf	HomB	HomE	HetB	HetE
Acc@1	0.028	0.201	0.192	0.224	<b>0.252</b>
Acc@5	0.159	0.313	0.369	0.393	<b>0.421</b>
Acc@10	0.243	0.346	0.407	0.439	<b>0.467</b>
Acc@20	0.313	0.411	0.467	0.50	<b>0.523</b>
Features		+ Glob	+ Glob	+ Glob	+ Glob
Acc@1		0.230	<b>0.285</b>	0.257	<b>0.285</b>
Acc@5		0.402	0.407	0.449	<b>0.458</b>
Acc@10		0.435	0.458	<b>0.50</b>	0.495
Acc@20		0.486	0.523	<b>0.565</b>	0.542

Table 4: The System Performance Based on Each Single Feature Set.

Features	Surf + HomB	Surf + HomE	Surf + HetB	Surf + HetE
Acc@1	0.234	0.238	0.262	<b>0.276</b>
Acc@5	0.416	0.444	0.481	<b>0.519</b>
Acc@10	0.477	0.505	0.533	<b>0.570</b>
Acc@20	0.519	0.561	0.565	<b>0.598</b>
Features	+ Glob	+ Glob	+ Glob	+ Glob
Acc@1	0.290	0.341	0.322	<b>0.346</b>
Acc@5	0.505	0.495	0.528	<b>0.533</b>
Acc@10	0.551	0.551	0.579	<b>0.584</b>
Acc@20	0.594	0.603	<b>0.636</b>	0.631

Table 5: The System Performance Based on Combinations of Surface and Semantic Features.

### 5.2.2 Cross Source and Cross Genre Information

We integrate the cross source information from Twitter, and the cross genre information from web documents into Weibo tweets for information network construction, and extract a new set of semantic features. Table 6 shows that further gains can be achieved. Notice that integrating tweets from Twitter mainly improves the ranking for top  $k$  where  $k > 1$ . This is because Weibo dominates our dataset, and in Weibo many of these sensitive morphs are mostly used with their traditional meanings instead of the morph senses. Further performance improvement is achieved by integrating information from background formal web documents which can provide richer context and relations.

Features	Surf + HomB + Glob	Surf + HomE + Glob	Surf + HetB + Glob	Surf + HetE + Glob
Acc@1	0.290	0.341	0.322	<b>0.346</b>
Acc@5	0.505	0.495	0.528	<b>0.533</b>
Acc@10	0.551	0.551	0.579	<b>0.584</b>
Acc@20	0.594	0.603	<b>0.636</b>	0.631
Features	+ Twit- ter	+ Twit- ter	+ Twit- ter	+ Twit- ter
Acc@1	0.308	0.336	0.336	<b>0.346</b>
Acc@5	0.514	0.519	0.547	<b>0.565</b>
Acc@10	0.579	0.594	0.594	<b>0.636</b>
Acc@20	0.631	0.640	<b>0.668</b>	<b>0.668</b>
Features	+ Web	+ Web	+ Web	+ Web
Acc@1	0.327	0.360	0.341	<b>0.379</b>
Acc@5	0.528	0.519	0.565	<b>0.575</b>
Acc@10	0.594	0.589	0.622	<b>0.645</b>
Acc@20	0.631	0.650	<b>0.678</b>	<b>0.678</b>

Table 6: The System Performance of Integrating Cross Source and Cross Genre Information.

### 5.2.3 Effects of Social Features

Table 7 shows that adding social features can improve the best performance achieved so far. This is because a group of people with close relationships may share similar opinion. As an example, two tweets “...of course the reputation of Buhou is a little too high! //@User1: //@User2: Chongqing event tells us...” and “...do not follow Bo Xilai...@User1...” are from two users in the same social group. One includes a morph “Buhou” and the other includes its target “Bo Xilai”.

Features	Surf + HomB + Glob + Twitter + Web	Surf + HomE + Glob + Twitter + Web	Surf + HetB + Glob + Twitter + Web	Surf + HetE + Glob + Twitter + Web
Acc@1	0.327	0.360	0.341	<b>0.379</b>
Acc@5	0.528	0.519	0.565	<b>0.575</b>
Acc@10	0.594	0.589	0.622	<b>0.645</b>
Acc@20	0.631	0.650	<b>0.678</b>	<b>0.678</b>
Features	+ Social	+ Social	+ Social	+ Social
Acc@1	0.336	0.369	0.365	<b>0.379</b>
Acc@5	0.537	0.547	0.589	<b>0.594</b>
Acc@10	0.594	0.601	0.645	<b>0.659</b>
Acc@20	0.645	0.664	<b>0.701</b>	<b>0.701</b>

Table 7: The Effects of Social Features.

### 5.3 Effects of Candidate Detection

The performance with and without candidate detection step (using all features) is shown in Table 8. The gain is small since the combination of all features in the learning to rank framework can already well capture the relationship between a morph and a target candidate. Nevertheless, the temporal distribution assumption is effective. It helps to filter out 80% of unrelated targets and

speed up the system 5 times, while retain 98.5% of the morph candidates that can be detected.

System	Acc@1	Acc@5	Acc@10	Acc@20
Without	0.365	0.579	0.645	0.696
With	<b>0.379</b>	<b>0.594</b>	<b>0.659</b>	<b>0.701</b>

Table 8: The Effects of Temporal Constraint

We also attempted using topic modeling approach to detect target candidates. Due to the large amount of data, we first split the data set on a daily basis, then applied Probabilistic Latent Semantic Analysis (PLSA) (Hofmann, 1999). Named entities which co-occur at least  $\delta$  times with a morph query in the same topic are selected as its target candidates. As shown in Table9 ( $K$  is the number of predefined topics), PLSA is not quite effective mainly because traditional topic modeling approaches do not perform well on short texts from social media. Therefore, in this paper we choose a simple method based on temporal distribution to detect target candidates.

Method	All	Temporal	PLSA( $K = 5$ $\delta = 1$ )	PLSA( $K = 5$ $\delta = 2$ )
Acc	0.935	0.921	0.935	0.925
No.	8,111	1,964	6,380	4,776
Method	PLSA( $K = 10$ $\delta = 1$ )	PLSA( $K = 10$ $\delta = 2$ )	PLSA( $K = 20$ $\delta = 1$ )	PLSA( $K = 20$ $\delta = 2$ )
Acc	0.935	0.907	0.888	0.757
No.	5,117	3,138	3,702	1,664

Table 9: Accuracy of Target Candidate Detection

### 5.4 Discussions

Compared with the standard alias detection (“Surf+HomB”) approach (Hsiung et al., 2005), our proposed approach achieves significantly better performance (99.9% confidence level by the Wilcoxon Matched-Pairs Signed-Ranks Test for Acc@1). We further explore two types of factors which may affect the system performance as follows.

One important aspect affecting the resolution performance is the morph & non-morph ambiguity. We categorize a morph query as “Unique” if the string is mainly used as a morph when it occurs, such as “薄督 (Bodu)” which is used to refer to “Bo Xilai”; otherwise as “Common” (e.g. “宝宝 (Baby)”, “校长 (President)”). Table 10 presents the separate scores for these two categories. We can see that the morphs in “Unique”

category have much better resolution performance than those in “Common” category.

Category	Number	Acc@1	Acc@5	Acc@10	Acc@20
Unique	72	0.479	0.715	0.771	0.819
Common	35	0.171	0.343	0.40	0.429

Table 10: Performance of Two Categories

We also investigate the effects of popularity of morphs on the resolution performance. We split the queries into 5 bins with equal size based on the non-descending frequency, and evaluate Acc@1 separately. As shown in Table 11, we can see that the popularity is not highly correlated with the performance.

Rank	0 ~ 20%	20% ~ 40%	40% ~ 60%	60% ~ 80%	80% ~ 100%
All	0.333	0.476	0.341	0.429	0.318
Unique	0.321	0.679	0.379	0.571	0.483
Common	0.214	0.214	0.071	0.071	0.286

Table 11: Effects of Popularity of Morphs

## 6 Related Work

To analyze social media behavior under active censorship, (Bamman et al., 2012) automatically discovered politically sensitive terms from Chinese tweets based on message deletion analysis. In contrast, our work goes beyond target identification by resolving implicit morphs to their real targets.

Our work is closely related to alias detection (Hsiung et al., 2005; Pantel, 2006; Bollegala et al., 2011; Holzer et al., 2005). We demonstrated that state-of-the-art alias detection methods did not perform well on morph resolution. In this paper we exploit cross-genre information and social correlation to measure semantic similarity. (Yang et al., 2011; Huang et al., 2012) also showed the effectiveness of exploiting information from formal web documents to enhance tweet summarization and tweet ranking.

Other similar research lines are the TAC-KBP Entity Linking (EL) (Ji et al., 2010; Ji et al., 2011), which links a named entity in news and web documents to an appropriate knowledge base (KB) entry, the task of mining name translation pairs from comparable corpora (Udupa et al., 2009; Ji, 2009; Fung and Yee, 1998; Rapp, 1999; Shao and Ng, 2004; Hassan et al., 2007) and the link prediction problem (Adamic and Adar, 2001; Liben-Nowell and Kleinberg, 2003; Sun et al., 2011b;

Hasan et al., 2006; Wang et al., 2007; Sun et al., 2011a). Most of the work focused on unstructured or structured data with clean and rich relations (e.g. DBLP). In contrast, our work constructs heterogeneous information networks from unstructured, noisy multi-genre text without explicit entity attributes.

## 7 Conclusion and Future Work

To the best of our knowledge, this is the first work of resolving implicit information morphs from the data under active censorship. Our promising results can well serve as a benchmark for this new problem. Both of the Meta-path based and social correlation based semantic similarity measurements are proven powerful and complementary.

In this paper we have focused on entity morphs. In the future we will extend our method to discover other types of information morphs, such as events and nominal mentions. In addition, automatic identification of candidate morphs is another challenging task, especially when the mentions are ambiguous and can also refer to other real entities. Our ongoing work includes identifying candidate morphs from scratch, as well as discovering morphs for a given target based on anomaly analysis and textual coherence modeling.

## Acknowledgments

Thanks to the three anonymous reviewers for their insightful comments. This work was supported by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. NSF CAREER Award under Grant IIS-0953149, the U.S. NSF EAGER Award under Grant No. IIS-1144111, the U.S. DARPA FA8750-13-2-0041 - Deep Exploration and Filtering of Text (DEFT) Program, the U.S. DARPA under Agreement No. W911NF-12-C-0028, CUNY Junior Faculty Award, NSF IIS-0905215, CNS-0931975, CCF-0905014, and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## References

- Lada A. Adamic and Eytan Adar. 2001. Friends and neighbors on the web. *SOCIAL NETWORKS*, 25:211–230.
- Aris Anagnostopoulos, Ravi Kumar, and Mohammad Mahdian. 2008. Influence and correlation in social networks. In *KDD*, pages 7–15.
- David Bamman, Brendan O’Connor, and Noah A. Smith. 2012. Censorship and deletion practices in chinese social media. *First Monday*, 17(3).
- Patrick Barwise and Seán Meehan. 2010. The one thing you must get right when building a brand. *Harvard Business Review*, 88(12):80–84.
- D. Bollegala, Y. Matsuo, and M. Ishizuka. 2011. Automatic discovery of personal name aliases from the web. *Knowledge and Data Engineering, IEEE Transactions on*, 23(6):831–844.
- Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing chinese word segmentation for machine translation performance. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT ’08, pages 224–232.
- Pascale Fung and Lo Yuen Yee. 1998. An ir approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL ’98, pages 414–420.
- Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. 2006. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*.
- Ahmed Hassan, Haytham Fahmy, and Hany Hassan. 2007. Improving named entity translation by exploiting comparable and parallel corpora. In *RANLP*.
- Daniel S. Hirschberg. 1977. Algorithms for the longest common subsequence problem. *J. ACM*, 24(4):664–675.
- Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR ’99, pages 50–57.
- Ralf Holzer, Bradley Malin, and Latanya Sweeney. 2005. Email alias detection using social network analysis. In *Conference on Knowledge Discovery in Data: Proceedings of the 3rd international workshop on Link discovery*, volume 21, pages 52–57.
- Paul Hsiung, Andrew Moore, Daniel Neill, and Jeff Schneider. 2005. Alias detection in link data sets. In *Proceedings of the International Conference on Intelligence Analysis*, May.
- Hongzhao Huang, Arkaitz Zubiaga, Heng Ji, Hongbo Deng, Dong Wang, Hieu Khac Le, Tarek F. Abdelzaher, Jiawei Han, Alice Leung, John Hancock, and Clare R. Voss. 2012. Tweet ranking based on heterogeneous networks. In *COLING*, pages 1239–1256.
- Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL*, pages 254–262.
- H. Ji, R. Grishman, H.T. Dang, K. Griffith, and J. Ellis. 2010. Overview of the tac 2010 knowledge base population track. In *Text Analysis Conference (TAC) 2010*.
- H. Ji, R. Grishman, and H.T. Dang. 2011. Overview of the tac 2011 knowledge base population track. In *Text Analysis Conference (TAC) 2011*.
- Heng Ji. 2009. Mining name translations from comparable corpora by creating bilingual information networks. In *Proceedings of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, BUCC ’09, pages 34–37.
- David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM ’03, pages 556–559.
- Ching-Yung Lin, Lynn Wu, Zhen Wen, Hanghang Tong, Vicky Griffiths-Fisher, Lei Shi, and David Lubensky. 2012. Social network analysis in enterprise. *Proceedings of the IEEE*, 100(9):2759–2776.
- Vincent Ng. 2010. Supervised noun phrase coreference research: the first fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL ’10, pages 1396–1411.
- Patrick Pantel. 2006. Alias detection in malicious environments. In *AAAI Fall Symposium on Capturing and Using Patterns for Evidence Detection*, pages 14–20.
- Reinhard Rapp. 1999. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL ’99, pages 519–526.
- Li Shao and Hwee Tou Ng. 2004. Mining new word translations from comparable corpora. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING ’04.
- Yizhou Sun, Rick Barber, Manish Gupta, Charu C. Aggarwal, and Han Jiawei. 2011a. Co-author relationship prediction in heterogeneous bibliographic networks. In *Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining*, ASONAM ’11, pages 121–128.

- Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011b. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *PVLDB*, 4(11):992–1003.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180.
- Raghavendra Udupa, K. Saravanan, A. Kumaran, and Jagadeesh Jagarlamudi. 2009. Mint: a method for effective and scalable mining of named entity transliterations from large comparable corpora. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 799–807.
- Robert A. Wagner and Michael J. Fischer. 1974. The string-to-string correction problem. *J. ACM*, 21(1):168–173.
- Chao Wang, Venu Satuluri, and Srinivasan Parthasarathy. 2007. Local probabilistic models for link prediction. In *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, ICDM '07, pages 322–331.
- Zhen Wen and Ching-Yung Lin. 2010. On the quality of inferring interests from social neighbors. In *KDD*, pages 373–382.
- Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhong Su, and Juanzi Li. 2011. Social context summarization. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 255–264.
- Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. Hhmm-based chinese lexical analyzer ictclas. In *Proceedings of the second SIGHAN workshop on Chinese language processing - Volume 17*, SIGHAN '03, pages 184–187.

# Learning to Extract International Relations from Political Context

**Brendan O’Connor**

School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA

brenocon@cs.cmu.edu

**Brandon M. Stewart**

Department of Government  
Harvard University  
Cambridge, MA 02139, USA

bstewart@fas.harvard.edu

**Noah A. Smith**

School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA

nasmith@cs.cmu.edu

## Abstract

We describe a new probabilistic model for extracting events between major political actors from news corpora. Our unsupervised model brings together familiar components in natural language processing (like parsers and topic models) with contextual political information—temporal and dyad dependence—to infer latent event classes. We quantitatively evaluate the model’s performance on political science benchmarks: recovering expert-assigned event class valences, and detecting real-world conflict. We also conduct a small case study based on our model’s inferences.

A supplementary appendix, and replication software/data are available online, at: <http://brenocon.com/irevents>

## 1 Introduction

The digitization of large news corpora has provided an unparalleled opportunity for the systematic study of international relations. Since the mid-1960s political scientists have used political *events data*, records of public micro-level interactions between major political actors of the form “someone does something to someone else” as reported in the open press (Schrodt, 2012), to study the patterns of interactions between political actors and how they evolve over time. Scaling this data effort to modern corpora presents an information extraction challenge: can a structured collection of accurate, politically relevant events between major political actors be extracted automatically and efficiently? And can they be grouped into meaningful event types with a low-dimensional structure useful for further analysis?

We present an unsupervised approach to event extraction, in which political structure and linguistic evidence are combined. A political context

model of the relationship between a pair of political actors imposes a prior distribution over types of linguistic events. Our probabilistic model infers latent frames, each a distribution over textual expressions of a kind of event, as well as a representation of the relationship between each political actor pair at each point in time. We use syntactic preprocessing and a logistic normal topic model, including latent temporal smoothing on the political context prior.

We apply the model in a series of comparisons to benchmark datasets in political science. First, we compare the automatically learned verb classes to a pre-existing ontology and hand-crafted verb patterns from TABARI,<sup>1</sup> an open-source and widely used rule-based event extraction system for this domain. Second, we demonstrate correlation to a database of real-world international conflict events, the Militarized Interstate Dispute (MID) dataset (Jones *et al.*, 1996). Third, we qualitatively examine a prominent case not included in the MID dataset, Israeli-Palestinian relations, and compare the recovered trends to the historical record.

We outline the data used for event discovery (§2), describe our model (§3), inference (§4), evaluation (§5), and comment on related work (§6).

## 2 Data

The model we describe in §3 is learned from a corpus of 6.5 million newswire articles from the English Gigaword 4th edition (1994–2008, Parker *et al.*, 2009). We also supplement it with a sample of data from the *New York Times* Annotated Corpus (1987–2007, Sandhaus, 2008).<sup>2</sup> The Stan-

<sup>1</sup>Available from the Penn State Event Data Project: <http://eventdata.psu.edu/>

<sup>2</sup>For arbitrary reasons this portion of the data is much smaller (we only parse the first five sentences of each article, while Gigaword has all sentences parsed), resulting in less than 2% as many tuples as from the Gigaword data.

ford CoreNLP system,<sup>3</sup> under default settings, was used to POS-tag and parse the articles, to eventually produce event tuples of the form

$$\langle s, r, t, w_{\text{predpath}} \rangle$$

where  $s$  and  $r$  denote “source” and “receiver” arguments, which are political actor entities in a pre-defined set  $\mathcal{E}$ ,  $t$  is a timestep (i.e., a 7-day period) derived from the article’s published date, and  $w_{\text{predpath}}$  is a textual predicate expressed as a dependency path that typically includes a verb (we use the terms “predicate-path” and “verb-path” interchangeably). For example, on January 1, 2000, the AP reported “Pakistan promptly accused India,” from which our preprocessing extracts the tuple  $\langle \text{PAK}, \text{IND}, 678, \text{accuse} \xrightarrow{\text{dobj}} \rangle$ . (The path excludes the first source-side arc.) Entities and verb paths are identified through the following sets of rules.

Named entity recognition and resolution is done deterministically by finding instances of country names from the *CountryInfo.txt* dictionary from TABARI,<sup>4</sup> which contains proper noun and adjectival forms for countries and administrative units. We supplement these with a few entries for international organizations from another dictionary provided by the same project, and clean up a few ambiguous names, resulting in a final actor dictionary of 235 entities and 2,500 names.

Whenever a name is found, we identify its entity’s mention as the minimal noun phrase that contains it; if the name is an adjectival or noun-noun compound modifier, we traverse any such *amod* and *nn* dependencies to the noun phrase head. Thus *NATO bombing*, *British view*, and *Palestinian militant* resolve to the entity codes IG-ONAT, GBR, and PSE respectively.

We are interested in identifying actions initiated by agents of one country targeted towards another, and hence concentrate on verbs, analyzing the “CCprocessed” version of the Stanford Dependencies (de Marneffe and Manning, 2008). Verb paths are identified by looking at the shortest dependency path between two mentions in a sentence. If one of the mentions is immediately dominated by a *nsubj* or *agent* relation, we consider that the Source actor, and the other mention is the Receiver. The most common cases are simple direct objects and prepositional arguments like *talk*

<sup>3</sup><http://nlp.stanford.edu/software/corenlp.shtml>

<sup>4</sup><http://eventdata.psu.edu/software/dir/dictionaries.html>.

$\xleftarrow{\text{prep.with}}$  and *fight*  $\xleftarrow{\text{prep.alongside}}$  (“talk with  $R$ ,” “fight alongside  $R$ ”) but many interesting multiword constructions also result, such as *reject*  $\xleftarrow{\text{dobj}}$  *allegation*  $\xleftarrow{\text{poss}}$  (“rejected  $R$ ’s allegation”) or verb chains as in *offer*  $\xleftarrow{\text{xcomp}}$  *help*  $\xleftarrow{\text{dobj}}$  (“offer to help  $R$ ”).

We wish to focus on instances of directly reported events, so attempt to remove factively complicated cases such as indirect reporting and hypotheticals by discarding all predicate paths for which any verb on the path has an off-path governing verb with a non-*conj* relation. (For example, the verb at the root of a sentence always survives this filter.) Without this filter, the  $\langle s, r, w \rangle$  tuple  $\langle \text{USA}, \text{CUB}, \text{want} \xleftarrow{\text{xcomp}} \text{seize} \xleftarrow{\text{dobj}} \rangle$  is extracted from the sentence “Parliament Speaker Ricardo Alarcon said the United States wants to seize Cuba and take over its lands”; the filter removes it since *wants* is dominated by an off-path verb through *say*  $\xleftarrow{\text{ccomp}}$  *wants*. The filter was iteratively developed by inspecting dozens of output examples and their labelings under successive changes to the rules.

Finally, only paths length 4 or less are allowed, the final dependency relation for the receiver may not be *nsubj* or *agent*, and the path may not contain any of the dependency relations *conj*, *parataxis*, *det*, or *dep*. We use lemmatized word forms in defining the paths.

Several document filters are applied before tuple extraction. Deduplication removes 8.5% of articles.<sup>5</sup> For topic filtering, we apply a series of keyword filters to remove sports and finance news, and also apply a text classifier for diplomatic and military news, trained on several hundred manually labeled news articles (using  $\ell_1$ -regularized logistic regression with unigram and bigram features). Other filters remove non-textual junk and non-standard punctuation likely to cause parse errors.

For experiments we remove tuples where the source and receiver entities are the same, and restrict to tuples with dyads that occur at least 500 times, and predicate paths that occur at least 10 times. This yields 365,623 event tuples from 235,830 documents, for 421 dyads and 10,457 unique predicate paths. We define timesteps to be 7-day periods, resulting in 1,149 discrete

<sup>5</sup>We use a simple form of shingling (ch. 3, Rajaraman and Ullman, 2011): represent a document signature as its  $J = 5$  lowercased bigrams with the lowest hash values, and reject a document with a signature that has been seen before within the same month.  $J$  was manually tuned, as it affects the precision/recall tradeoff.

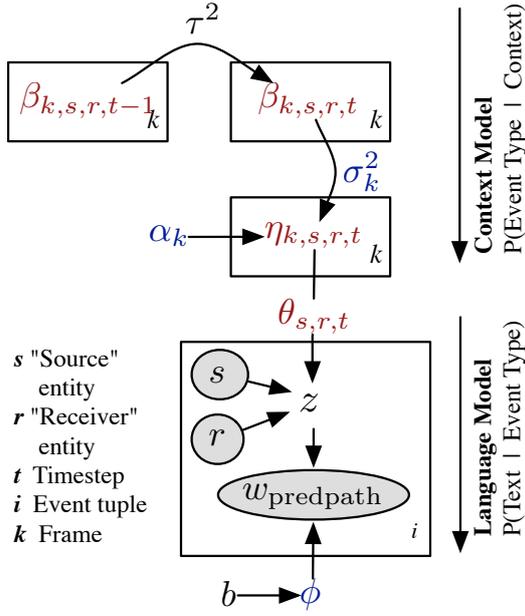


Figure 1: Directed probabilistic diagram of the model for one  $(s, r, t)$  dyad-time context, for the smoothed model.

timesteps (1987 through 2008, though the vast majority of data starts in 1994).

### 3 Model

We design two models to learn linguistic event classes over predicate paths by conditioning on real-world contextual information about international politics,  $p(w_{\text{predpath}} | s, r, t)$ , leveraging the fact there tends to be dyadic and temporal coherence in international relations: the types of actions that are likely to occur between nations tend to be similar within the same dyad, and usually their distribution changes smoothly over time.

Our model decomposes into two submodels: a Context submodel, which encodes how political context affects the probability distribution over event types, and a Language submodel, for how those events are manifested as textual predicate paths (Figure 1). The overall generative process is as follows. We color global parameters for a frame **blue**, and local context parameters **red**, and use the term “frame” as a synonym for “event type.” The fixed hyperparameter  $K$  denotes the number of frames.

- The **context model** generates a frame prior  $\theta_{s,r,t}$  for every context  $(s, r, t)$ .
- **Language model:**
  - Draw lexical sparsity parameter  $b$  from a diffuse prior (see §4).

- For each frame  $k$ , draw a multinomial distribution of dependency paths,  $\phi_k \sim \text{Dir}(b/V)$  (where  $V$  is the number of dependency path types).
- For each  $(s, r, t)$ , for every event tuple  $i$  in that context,
  - Sample its frame  $z^{(i)} \sim \text{Mult}(\theta_{s,r,t})$ .
  - Sample its predicate realization  $w_{\text{predpath}}^{(i)} \sim \text{Mult}(\phi_{z^{(i)}})$ .

Thus the language model is very similar to a topic model’s generation of token topics and wordtypes.

We use structured logistic normal distributions to represent contextual effects. The simplest is the **vanilla (v)** context model,

- For each frame  $k$ , draw global parameters from diffuse priors: prevalence  $\alpha_k$  and variability  $\sigma_k^2$ .
- For each  $(s, r, t)$ ,
  - Draw  $\eta_{k,s,r,t} \sim N(\alpha_k, \sigma_k^2)$  for each frame  $k$ .
  - Apply a softmax transform,

$$\theta_{k,s,r,t} = \frac{\exp \eta_{k,s,r,t}}{\sum_{k'=1}^K \exp \eta_{k',s,r,t}}$$

Thus the vector  $\eta_{*,s,r,t}$  encodes the relative log-odds of the different frames for events appearing in the context  $(s, r, t)$ . This simple logistic normal prior is, in terms of topic models, analogous to the asymmetric Dirichlet prior version of LDA in Wallach *et al.* (2009), since the  $\alpha_k$  parameter can learn that some frames tend to be more likely than others. The variance parameters  $\sigma_k^2$  control admixture sparsity, and are analogous to a Dirichlet’s concentration parameter.

#### Smoothing Frames Across Time

The vanilla model is capable of inducing frames through dependency path co-occurrences, when multiple events occur in a given context. However, many dyad-time slices are very sparse; for example, most dyads (all but 18) have events in fewer than half the time slices in the dataset. One solution is to increase the bucket size (e.g., to months); however, previous work in political science has demonstrated that answering questions of interest about reciprocity dynamics requires recovering the events at weekly or even daily granularity (Shellman, 2004), and in any case wide buckets help only so much for dyads with fewer events or less media attention. Therefore we propose a **smoothed frames (SF)** model, in which the

frame distribution for a given dyad comes from a latent parameter  $\beta_{*,s,r,t}$  that smoothly varies over time. For each  $(s, r)$ , draw the first timestep’s values as  $\beta_{k,s,r,1} \sim N(0, 100)$ , and for each context  $(s, r, t > 1)$ ,

- Draw  $\beta_{k,s,r,t} \sim N(\beta_{k,s,r,t-1}, \tau^2)$
- Draw  $\eta_{k,s,r,t} \sim N(\alpha_k + \beta_{k,s,r,t}, \sigma_k^2)$

Other parameters  $(\alpha_k, \sigma_k^2)$  are same as the vanilla model. This model assumes a random walk process on  $\beta$ , a variable which exists even for contexts that contain no events. Thus inferences about  $\eta$  will be smoothed according to event data at nearby timesteps. This is an instance of a linear Gaussian state-space model (also known as a linear dynamical system or dynamic linear model), and is a convenient formulation because it has well-known exact inference algorithms. Dynamic linear models have been used elsewhere in machine learning and political science to allow latent topic frequencies (Blei and Lafferty, 2006; Quinn *et al.*, 2010) and ideological positions (Martin and Quinn, 2002) to smoothly change over time, and thus share statistical strength between timesteps.

## 4 Inference

After randomly initializing all  $\eta_{k,s,r,t}$ , inference is performed by a blocked Gibbs sampler, alternating resamplings for three major groups of variables: the language model  $(z, \phi)$ , context model  $(\alpha, \gamma, \beta, p)$ , and the  $\eta, \theta$  variables, which bottleneck between the submodels.

The **language model** sampler sequentially updates every  $z^{(i)}$  (and implicitly  $\phi$  via collapsing) in the manner of Griffiths and Steyvers (2004):  $p(z^{(i)} | \theta, w^{(i)}, b) \propto \theta_{s,r,t,z} (n_{w,z} + b/V) / (n_z + b)$ , where counts  $n$  are for all event tuples besides  $i$ .

For the **context model**,  $\alpha$  is conjugate resampled as a normal mean. The random walk variables  $\beta$  are sampled with the forward-filtering-backward-sampling algorithm (FFBS; Harrison and West, 1997; Carter and Kohn, 1994); there is one slight modification of the standard dynamic linear model that the zero-count weeks have no  $\eta$  observation; the Kalman filter implementation is appropriately modified to handle this.

The  $\eta$  update step is challenging since it is a nonconjugate prior to the  $z$  counts. Logistic normal distributions were introduced to text modeling by Blei and Lafferty (2007), who developed a variational approximation; however, we

find that experimenting with different models is easier in the Gibbs sampling framework. While Gibbs sampling for logistic normal priors is possible using auxiliary variable methods (Mimno *et al.*, 2008; Holmes and Held, 2006; Polson *et al.*, 2012), it can be slow to converge. We opt for the more computationally efficient approach of Zeger and Karim (1991) and Hoff (2003), using a Laplace approximation to  $p(\eta | \bar{\eta}, \Sigma, z)$ , which is a mode-centered Gaussian having inverse covariance equal to the unnormalized log-posterior’s negative Hessian (§8.4 in Murphy, 2012). We find the mode with the linear-time Newton algorithm from Eisenstein *et al.* (2011), and sample in linear time by only using the Hessian’s diagonal as the inverse covariance (i.e., an axis-aligned normal), since a full multivariate normal sample requires a cubic-time-to-compute Cholesky root of the covariance matrix. This  $\eta^*$  sample is a proposal for a Metropolis-within-Gibbs step, which is moved to according to the standard Metropolis-Hastings acceptance rule. Acceptance rates differ by  $K$ , ranging approximately from 30% ( $K = 100$ ) to nearly 100% (small  $K$ ).

Finally, we use diffuse priors on all global parameters, conjugate resampling variances  $\tau^2, \sigma_k$  once per iteration, and slice sampling (Neal, 2003) the Dirichlet concentration  $b$  every 100 iterations. Automatically learning these was extremely convenient for model-fitting; the only hyperparameter we set manually was  $K$ . It also allowed us to monitor the convergence of dispersion parameters to help debug and assess MCMC mixing. For other modeling and implementation details, see the online appendix and software.

## 5 Experiments

We fit the two models on the dataset described in §2, varying the number of frames  $K$ , with 8 or more separate runs for each setting. Posteriors are saved and averaged from 11 Gibbs samples (every 100 iterations from 9,000 to 10,000) for analysis.

We present intrinsic (§5.1) and extrinsic (§5.2) quantitative evaluations, and a qualitative case study (§5.4).

### 5.1 Lexical Scale Impurity

In the international relations literature, much of the analysis of text-based events data makes use of a unidimensional conflict to cooperation scale. A popular event ontology in this domain, CAMEO, consists of around 300 different event types, each

given an expert-assigned scale in the range from  $-10$  to  $+10$  (Gerner *et al.*, 2002), derived from a judgement collection experiment in Goldstein (1992). The TABARI pattern-based event extraction program comes with a list of almost 16,000 manually engineered verb patterns, each assigned to one CAMEO event type.

It is interesting to consider the extent to which our unsupervised model is able to recover the expert-designed ontology. Given that many of the categories are very fine-grained (e.g. “Express intent to de-escalate military engagement”), we elect to measure model quality as *lexical scale purity*: whether all the predicate paths within one automatically learned frame tend to have similar gold-standard scale scores. (This measures cluster cohesiveness against a one-dimensional continuous scale, instead of measuring cluster cohesiveness against a gold-standard clustering as in VI, Rand index, or purity.) To calculate this, we construct a mapping between our corpus-derived verb path vocabulary and the TABARI verb patterns, many of which contain one to several word stems that are intended to be matched in surface order. Many of our dependency paths, when traversed from the source to receiver direction, also follow surface order, due to English’s SVO word order.<sup>6</sup> Therefore we convert each path to a word sequence and match against the TABARI lexicon—plus a few modifications for differences in infinitives and stemming—and find 528 dependency path matches. We assign each path  $w$  a gold-standard scale  $g(w)$  by resolving through its matching pattern’s CAMEO code.

We formalize *lexical scale impurity* as the average absolute difference of scale values between two predicate paths under the same frame. Specifically, we want a token-level posterior expectation

$$\mathbb{E}(|g(w_i) - g(w_j)| \mid z_i = z_j, w_i \neq w_j) \quad (1)$$

which is taken over pairs of path instances  $(i, j)$  where both paths  $w_i, w_j$  are in  $M$ , the set of verb paths that were matched between the lexicons. This can be reformulated at the type level as:<sup>7</sup>

$$\frac{1}{N} \sum_k \sum_{\substack{w, v \in M \\ w \neq v}} n_{w,k} n_{v,k} |g(w) - g(v)| \quad (2)$$

<sup>6</sup>There are plenty of exceptions where a Source-to-Receiver path traversal can have a right-to-left move, such as dependency edges for possessives. This approach can not match them.

<sup>7</sup>Derivation in supplementary appendix.

where  $n$  refers to the averaged Gibbs samples’ counts of event tuples having frame  $k$  and a particular verb path,<sup>8</sup> and  $N$  is the number of token comparisons (i.e. the same sum, but with a 1 replacing the distance). The worst possible impurity is upper bounded at 20 ( $= \max(g(w)) - \min(g(w))$ ) and the best possible is 0. We also compute a randomized null hypothesis to see how low impurity can be by chance: each of  $\sim 1000$  simulations randomly assigns each path in  $M$  to one of  $K$  frames (all its instances are exclusively assigned to that frame), and computes the impurity. On average the impurity is same at all  $K$ , but variance increases with  $K$  (since small clusters might by chance get a highly similar paths in them), necessitating this null hypothesis analysis. We report the 5th percentile over simulations.

## 5.2 Conflict Detection

Political events data has shown considerable promise as a tool for crisis early warning systems (O’Brien, 2010; Brandt *et al.*, 2011). While conflict forecasting is a potential application of our model, we conduct a simpler prediction task to validate whether the model is learning something useful: based on news text, tell whether or not an armed conflict is *currently* happening. For a gold standard, we use the Militarized Interstate Dispute (MID) dataset (Jones *et al.*, 1996; Ghosn *et al.*, 2004), which documents historical international disputes. While not without critics, the MID data is the most prominent dataset in the field of international relations. We use the Dyadic MIDs, each of which ranks hostility levels between pairs of actors on a five point scale over a date interval; we define conflict to be the top two categories “Use of Force” (4) and “War” (5). We convert the data into a variable  $y_{s,r,t}$ , the highest hostility level reached by actor  $s$  directed towards receiver  $r$  in the dispute that overlaps with our 7-day interval  $t$ , and want to predict the binary indicator  $\mathbf{1}\{y_{s,r,t} \geq 4\}$ . For the illustrative examples (USA to Iraq, and the Israel-Palestine example below) we use results from a smaller but more internally comparable dataset consisting of the 2 million Associated Press articles within the Gigaword corpus.

For an example of the MID data, see Figure 2, which depicts three disputes between the US and

<sup>8</sup>Results are nearly identical whether we use counts averaged across samples (thus giving posterior marginals), or simply use counts from a single sample (i.e., iteration 10,000).

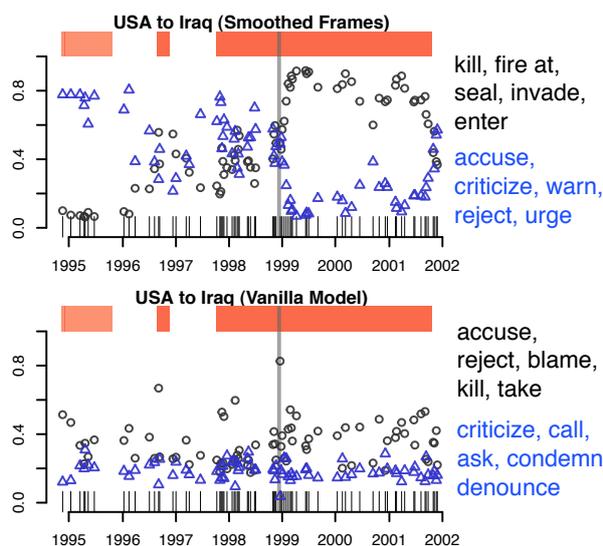


Figure 2: The USA→Iraq directed dyad, analyzed by smoothed (above) and vanilla (below) models, showing (1) gold-standard MID values (red intervals along top), (2) weeks with non-zero event counts (vertical lines along x-axis), (3) posterior  $E[\theta_{k, \text{USA}, \text{IRQ}, t}]$  inferences for two frames chosen from two different  $K = 5$  models, and (4) most common verb paths for each frame (right). Frames corresponding to material and verbal conflict were chosen for display. Vertical line indicates Operation Desert Fox (see §5.2).

Iraq in this time period. The MID labels are marked in red.

The first dispute is a “display of force” (level 3), cataloguing the U.S. response to a series of troop movements along the border with Kuwait. The third dispute (10/7/1997 to 10/10/2001) begins with increasing Iraqi violations of the no-fly zone, resulting in U.S. and U.K. retaliation, reaching a high intensity with Operation Desert Fox, a four-day bombing campaign from December 16 to 19, 1998—which is not shown in MID. These cases highlight MID’s limitations—while it is well regarded in the political science literature, its coarse level of aggregation can fail to capture variation in conflict intensity.

Figure 2 also shows model inferences. Our smoothed model captures some of these phenomena here, showing clear trends for two relevant frames, including a dramatic change in December 1998. The vanilla model has a harder time, since it cannot combine evidence between different timesteps.

The MID dataset overlaps with our data for 470 weeks, from 1993 through 2001. After excluding dyads with actors that the MID data does not intend to include—Kosovo, Tibet, Palestine, and international organizations—we have 267 directed dyads for evaluation, 117 of which have at least

one dispute in the MID data. (Dyads with no dispute in the MID data, such as Germany-France, are assumed to have  $y = 0$  throughout the time period.) About 7% of the dyad-time contexts have a dispute under these definitions.

We split the dataset by time, training on the first half of the data and testing on the second half, and measure area under the receiver operating characteristic curve (AUC).<sup>9</sup> For each model, we train an  $\ell_1$ -regularized logistic regression<sup>10</sup> with the  $K$  elements of  $\theta_{*,s,r,t}$  as input features, tuning the regularization parameter within the training set (by splitting it in half again) to optimize held-out likelihood. We weight instances to balance positive and negative examples. Training is on all individual  $\theta$  samples at once (thus accounting for posterior uncertainty in learning), and final predicted probabilities are averaged from individual probabilities from each  $\theta$  test set sample, thus propagating posterior uncertainty into the predictions. We also create a baseline  $\ell_1$ -regularized logistic regression that uses normalized dependency path counts as the features (10,457 features). For both the baseline and vanilla model, contexts with no events are given a feature vector of all zeros.<sup>11</sup> (We also explored an alternative evaluation setup, to hold out by dyad; however, the performance variance is quite high between different random dyad splits.)

### 5.3 Results

Results are shown in Figure 3.<sup>12</sup>

The verb-path logistic regression performs strongly at AUC 0.62; it outperforms all of the vanilla frame models. This is an example of individual lexical features outperforming a topic model for predictive task, because the topic model’s dimension reduction obscures important indicators from individual words. Similarly, [Gerish and Blei \(2011\)](#) found that word-based regression outperformed a customized topic model when predicting Congressional bill passage, and [Eisen-](#)

<sup>9</sup>AUC can be interpreted as follows: given a positive and negative example, what is the probability that the classifier’s confidences order them correctly? Random noise or predicting all the same class both give AUC 0.5.

<sup>10</sup>Using the R *glmnet* package (Friedman *et al.*, 2010).

<sup>11</sup>For the vanilla model, this performed better than linear interpolation (about 0.03 AUC), and with less variance between runs.

<sup>12</sup>Due to an implementation bug, the model put the vast majority of the probability mass only on  $K - 1$  frames, so these settings might be better thought of as  $K = 1, 2, 3, 4, 9, \dots$ ; see the appendix for details.

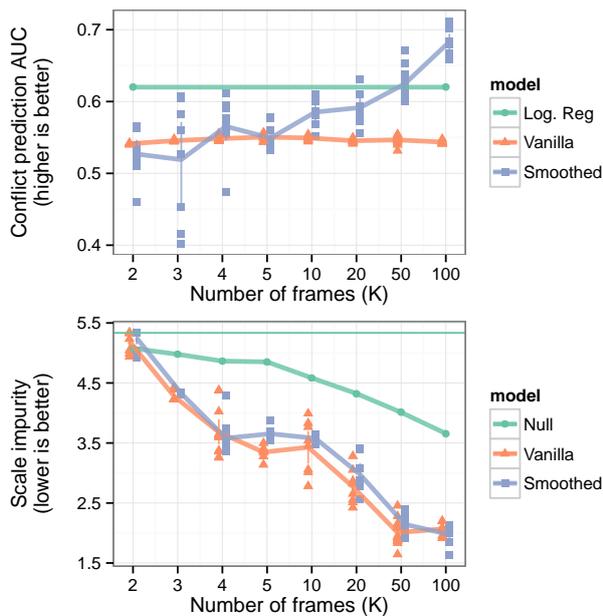


Figure 3: Evaluation results. Each point indicates one model run. Lines show the average per  $K$ , with vertical lines indicating the 95% bootstrapped interval. **Top:** Conflict detection AUC for different models (§5.2). Green line is the verb-path logistic regression baseline. **Bottom:** Lexical scale impurity (§5.1). Top green line indicates the simple random baseline  $E(|g(w_i) - g(w_j)|) = 5.33$ ; the second green line is from the random assignment baseline.

stein *et al.* (2010) found word-based regression outperformed Supervised LDA for geolocation,<sup>13</sup> and we have noticed this phenomenon for other text-based prediction problems.

However, adding smoothing to the model substantially increases performance, and in fact outperforms the verb-path regression at  $K = 100$ . It is unclear why the vanilla model fails to increase performance in  $K$ . Note also, the vanilla model exhibits very little variability in prediction performance between model runs, in comparison to the smoothed model which is much more variable (presumably due to the higher number of parameters in the model); at small values of  $K$ , the smoothed model can perform poorly. It would also be interesting to analyze the smoothed model with higher values of  $K$  and find where it peaks.

We view the conflict detection task only as one of several validations, and thus turn to lexical evaluation of the induced frames. For lexical scale purity (bottom of Figure 3), the models perform about the same, with the smoothed model a little bit worse at some values of  $K$  (though sometimes with better stability of the fits—opposite of the conflict detection task). This suggests that semantic coherence does not benefit from the longer-

<sup>13</sup>In the latter, a problem-specific topic model did best.

range temporal dependencies.

In general, performance improves with higher  $K$ , but not beyond  $K = 50$ . This suggests the model reaches a limit for how fine-grained of semantics it can learn.

## 5.4 Case study

Here we qualitatively examine the narrative story between the dyad with the highest frequency of events in our dataset, the Israeli-Palestinian relationship, finding qualitative agreement with other case studies of this conflict (Brandt *et al.*, 2012; Goldstein *et al.*, 2001; Schrodt and Gerner, 2004). (The MID dataset does not include this conflict because the Palestinians are not considered a state actor.) Using the Associated Press subset, we plot the highest incidence frames from one run of the  $K = 20$  smoothed frame models, for the two directed dyads, and highlight some of the interesting relationships.

Figure 4(a) shows that tradeoffs in the use of military vs. police action by Israel towards the Palestinians tracks with major historical events. The first period in the data where police actions (‘impose, seal, capture, seize, arrest’) exceed military actions (‘kill, fire, enter, attack, raid’) is with the signing of the “Interim Agreement on the West Bank and the Gaza Strip,” also known as the Oslo II agreement. This balance persists until the abrupt breakdown in relations that followed the unsuccessful Camp David Summit in July of 2000, which generally marks the starting point of the wave of violence known as the Second Intifada.

In Figure 4(b) we show that our model produces a frame which captures the legal aftermath of particular events (‘accuse, criticize,’ but also ‘detain, release, extradite, charge’). Each of the major spikes in the data coincides with a particular event which either involves the investigation of a particular attack or series of attacks (as in A,B,E) or a discussion about prisoner swaps or mass arrests (as in events D, F, J).

Our model also picks up positive diplomatic events, as seen in Figure 4(c), a frame describing Israeli diplomatic actions towards Palestine (‘meet with, sign with, praise, say with, arrive in’). Not only do the spikes coincide with major peace treaties and negotiations, but the model correctly characterizes the relative lack of positively valenced action from the beginning of the Second Intifada until its end around 2005–2006.

In Figure 4(d) we show the relevant frames de-

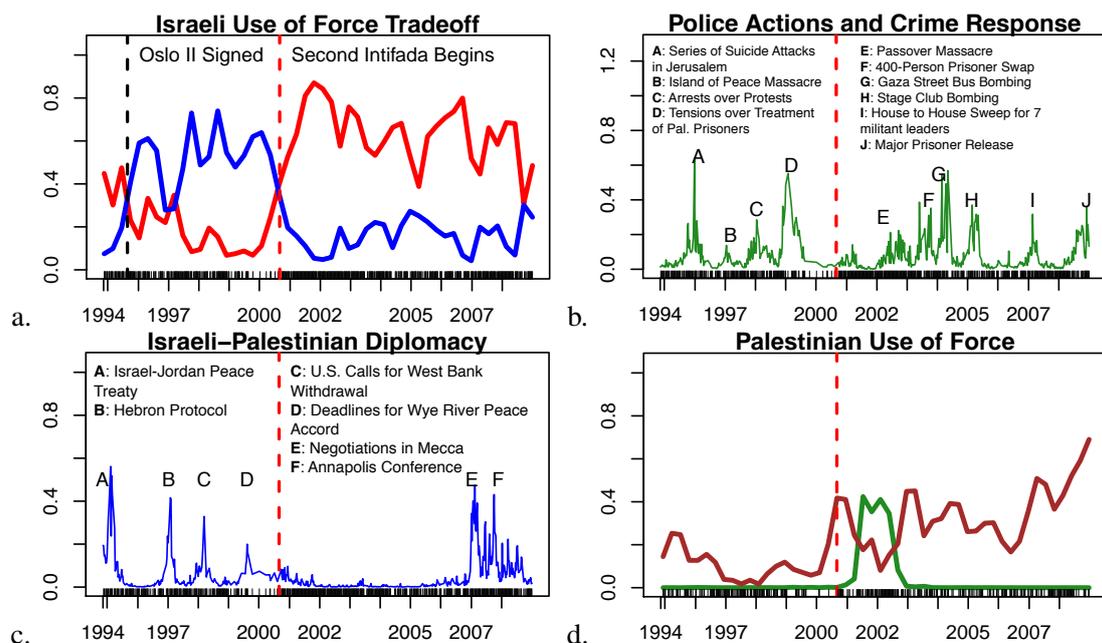


Figure 4: For Israel-Palestinian directed dyads, plots of  $E[\theta]$  (proportion of weekly events in a frame) over time, annotated with historical events. (a): Words are ‘kill, fire at, enter, kill, attack, raid, strike, move, pound, bomb’ and ‘impose, seal, capture, seize, arrest, ease, close, deport, close, release’ (b): ‘accuse, criticize, reject, tell, hand to, warn, ask, detain, release, order’ (c): ‘meet with, sign with, praise, say with, arrive in, host, tell, welcome, join, thank’ (d): again the same ‘kill, fire at’ frame in (a), plus the erroneous frame (see text) ‘include, join, fly to, have relation with, protest to, call, include bomber <sup>←appos</sup> informer for’. Figures (b) and (c) use linear interpolation for zero-count weeks (thus relying exclusively on the model for smoothing); (a) and (d) apply a loess smoother. (a-c) are for the ISR→PSE direction; (d) is PSE→ISR.

picting use of force from the Palestinians towards the Israelis (brown trend line). At first, the drop in the use of force frame immediately following the start of the Second Intifada seems inconsistent with the historical record. However, there is a concurrent rise in a different frame driven by the word ‘include’, which actually appears here due to an NLP error compounded with an artifact of the data source. A casualties report article, containing variants of the text “The Palestinian figure includes... 13 Israeli Arabs...”, is repeated 27 times over two years. “Palestinian figure” is erroneously identified as the PSE entity, and several noun phrases in a list are identified as separate receivers. This issue causes 39 of all 86 PSE→ISR events during this period to use the word ‘include’, accounting for the rise in that frame. (This highlights how better natural language processing could help the model, and the dangers of false positives for this type of data analysis, especially in small-sample drilldowns.) Discounting this erroneous inference, the results are consistent with heightened violence during this period.

We conclude the frame extractions for the Israeli-Palestinian case are consistent with the historical record over the period of study.

## 6 Related Work

### 6.1 Events Data in Political Science

Projects using hand-collected events data represent some of the earliest efforts in the statistical study of international relations, dating back to the 1960s (Rummel, 1968; Azar and Sloan, 1975; McClelland, 1970). Beginning in the mid-1980s, political scientists began experimenting with automated rule-based extraction systems (Schrodt and Gerner, 1994). These efforts culminated in the open-source program, TABARI, which uses pattern matching from extensive hand-developed phrase dictionaries, combined with basic part of speech tagging (Schrodt, 2001); a rough analogue in the information extraction literature might be the rule-based, finite-state FASTUS system for MUC IE (Hobbs *et al.*, 1997), though TABARI is restricted to single sentence analysis. Later proprietary work has apparently incorporated more extensive NLP (e.g., sentence parsing) though few details are available (King and Lowe, 2003). The most recent published work we know of, by Boschee *et al.* (2013), uses a proprietary parsing and coreference system (BBN SERIF, Ramshaw *et al.*, 2011), and directly compares to TABARI, finding significantly higher accuracy. The origi-

nal TABARI system is still actively being developed, including just-released work on a new 200 million event dataset, GDELT (Schrodtt and Leertaru, 2013).<sup>14</sup> All these systems crucially rely on hand-built pattern dictionaries.

It is extremely labor intensive to develop these dictionaries. Schrodtt (2006) estimates 4,000 trained person-hours were required to create dictionaries of political actors in the Middle East, and the phrase dictionary took dramatically longer; the comments in TABARI's phrase dictionary indicate some of its 15,789 entries were created as early as 1991. Ideally, any new events data solution would incorporate the extensive work already completed by political scientists in this area while minimizing the need for further dictionary development. In this work we use the actor dictionaries, and hope to incorporate the verb patterns in future work.

## 6.2 Events in Natural Language Processing

Political event extraction from news has also received considerable attention within natural language processing in part due to government-funded challenges such as MUC-3 and MUC-4 (Lehnert, 1994), which focused on the extraction of terrorist events, as well as the more recent ACE program. The work in this paper is inspired by unsupervised approaches that seek to discover types of relations and events, instead of assuming them to be pre-specified; this includes research under various headings such as template/frame/event learning (Cheung *et al.*, 2013; Modi *et al.*, 2012; Chambers and Jurafsky, 2011; Li *et al.*, 2010; Bejan, 2008), script learning (Regneri *et al.*, 2010; Chambers and Jurafsky, 2009), relation learning (Yao *et al.*, 2011), open information extraction (Banko *et al.*, 2007; Carlson *et al.*, 2010), verb caseframe learning (Rooth *et al.*, 1999; Gildea, 2002; Grenager and Manning, 2006; Lang and Lapata, 2010; Ó Séaghdha, 2010; Titov and Klementiev, 2012), and a version of frame learning called "unsupervised semantic parsing" (Titov and Klementiev, 2011; Poon and Domingos, 2009). Unlike much of the previous literature, we do not learn latent roles/slots. Event extraction is also a large literature, including supervised systems targeting problems similar to MUC and political events (Piskorski and Atkinson, 2011; Piskorski *et al.*, 2011; Sanfilippo *et al.*, 2008).

One can also see this work as a relational ex-

<sup>14</sup><http://eventdata.psu.edu/data.dir/GDELT.html>

ension of co-occurrence-based methods such as Gerrish (2013; ch. 4), Diesner and Carley (2005), Chang *et al.* (2009), or Newman *et al.* (2006), which perform bag-of-words-style analysis of text fragments containing co-occurring entities. (Gerrish also analyzed the international relations domain, using supervised bag-of-words regression to assess the expressed valence between a pair of actors in a news paragraph, using the predictions as observations in a latent temporal model, and compared to MID.) We instead use parsing to get a much more focused and interpretable representation of the relationship between textually co-occurring entities; namely, that they are the source and target of an action event. This is more in line with work in relation extraction on biomedical scientific articles (Friedman *et al.*, 2001; Rzhetsky *et al.*, 2004) which uses parsing to extracting a network of how different entities, like drugs or proteins, interact.

## 7 Conclusion

Large-scale information extraction can dramatically enhance the study of political behavior. Here we present a novel unsupervised approach to an important data collection effort in the social sciences. We see international relations as a rich and practically useful domain for the development of text analysis methods that jointly infer events, relations, and sociopolitical context. There are numerous areas for future work, such as: using verb dictionaries as semi-supervised seeds or priors; interactive learning between political science researchers and unsupervised algorithms; building low-dimensional scaling, or hierarchical structure, into the model; and learning the actor lists to handle changing real-world situations and new domains. In particular, adding more supervision to the model will be crucial to improve semantic quality and make it useful for researchers.

## Acknowledgments

Thanks to Justin Betteridge for providing the parsed Gigaword corpus, Erin Baggott for help in developing the document filter, and the anonymous reviewers for helpful comments. This research was supported in part by NSF grant IIS-1211277, and was made possible through the use of computing resources made available by the Pittsburgh Supercomputing Center. Brandon Stewart gratefully acknowledges funding from an NSF Graduate Research Fellowship.

## References

Azar, E. E. and Sloan, T. (1975). Dimensions of interactions. Technical report, University Center of International Studies, University of Pittsburgh, Pittsburgh.

- Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open Information Extraction from the Web. *IJCAI*.
- Bejan, C. A. (2008). Unsupervised discovery of event scenarios from texts. In *Proceedings of the 21st Florida Artificial Intelligence Research Society International Conference (FLAIRS)*, Coconut Grove, FL, USA.
- Blei, D. M. and Lafferty, J. D. (2006). Dynamic topic models. In *Proceedings of ICML*.
- Blei, D. M. and Lafferty, J. D. (2007). A correlated topic model of science. *Annals of Applied Statistics*, **1**(1), 17–35.
- Boschee, E., Natarajan, P., and Weischedel, R. (2013). Automatic extraction of events from open source text for predictive forecasting. *Handbook of Computational Approaches to Counterterrorism*, page 51.
- Brandt, P. T., Freeman, J. R., and Schrodt, P. A. (2011). Real time, time series forecasting of inter-and intra-state political conflict. *Conflict Management and Peace Science*, **28**(1), 41–64.
- Brandt, P. T., Freeman, J. R., Lin, T.-m., and Schrodt, P. A. (2012). A Bayesian time series approach to the comparison of conflict dynamics. In *APSA 2012 Annual Meeting Paper*.
- Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E. R., and Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 1306–1313.
- Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models. *Biometrika*, **81**(3), 541–553.
- Chambers, N. and Jurafsky, D. (2009). Unsupervised learning of narrative schemas and their participants. In *Proceedings of ACL-IJCNLP*. Association for Computational Linguistics.
- Chambers, N. and Jurafsky, D. (2011). Template-based information extraction without the templates. In *Proceedings of ACL*.
- Chang, J., Boyd-Graber, J., and Blei, D. M. (2009). Connections between the lines: augmenting social networks with text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178. ACM.
- Cheung, J. C. K., Poon, H., and Vanderwende, L. (2013). Probabilistic frame induction. In *Proceedings of NAACL*. arXiv preprint arXiv:1302.4813.
- de Marneffe, M.-C. and Manning, C. D. (2008). Stanford typed dependencies manual. Technical report, Stanford University.
- Diesner, J. and Carley, K. M. (2005). Revealing social structure from texts: meta-matrix text analysis as a novel method for network text analysis. In *Causal mapping for information systems and technology research*, pages 81–108. Harrisburg, PA: Idea Group Publishing.
- Eisenstein, J., O’Connor, B., Smith, N. A., and Xing, E. P. (2010). A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287.
- Eisenstein, J., Ahmed, A., and Xing, E. (2011). Sparse additive generative models of text. In *Proceedings of ICML*, pages 1041–1048.
- Friedman, C., Kra, P., Yu, H., Krauthammer, M., and Rzhetsky, A. (2001). GENIES: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, **17**(suppl 1), S74–S82.
- Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, **33**(1).
- Gerner, D. J., Schrodt, P. A., Yilmaz, O., and Abu-Jabr, R. (2002). The Creation of CAMEO (Conflict and Mediation Event Observations): An Event Data Framework for a Post Cold War World. *Annual Meeting of the American Political Science Association*.
- Gerrish, S. M. (2013). *Applications of Latent Variable Models in Modeling Influence and Decision Making*. Ph.D. thesis, Princeton University.
- Gerrish, S. M. and Blei, D. M. (2011). Predicting legislative roll calls from text. In *Proceedings of ICML*.
- Ghosn, F., Palmer, G., and Bremer, S. A. (2004). The MID3 data set, 1993–2001: Procedures, coding rules, and description. *Conflict Management and Peace Science*, **21**(2), 133–154.
- Gildea, D. (2002). Probabilistic models of verb-argument structure. In *Proceedings of COLING*.
- Goldstein, J. S. (1992). A conflict-cooperation scale for WEIS events data. *Journal of Conflict Resolution*, **36**, 369–385.
- Goldstein, J. S., Pevehouse, J. C., Gerner, D. J., and Telhami, S. (2001). Reciprocity, triangularity, and cooperation in the middle east, 1979-97. *Journal of Conflict Resolution*, **45**(5), 594–620.
- Grenager, T. and Manning, C. D. (2006). Unsupervised discovery of a statistical verb lexicon. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, page 18.
- Griffiths, T. L. and Steyvers, M. (2004). Finding scientific topics. *PNAS*, **101**(suppl. 1), 5228–5235.
- Harrison, J. and West, M. (1997). *Bayesian forecasting and dynamic models*. Springer Verlag, New York.
- Hobbs, J. R., Appelt, D., Bear, J., Israel, D., Kameyama, M., Stickel, M., and Tyson, M. (1997). FASTUS: A cascaded finite-state transducer for extracting information from natural-language text. *Finite-State Language Processing*, page 383.
- Hoff, P. D. (2003). Nonparametric modeling of hierarchically exchangeable data. *University of Washington Statistics Department, Technical Report*, **421**.
- Holmes, C. C. and Held, L. (2006). Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, **1**(1), 145–168.
- Jones, D., Bremer, S., and Singer, J. (1996). Militarized interstate disputes, 1816–1992: Rationale, coding rules, and empirical patterns. *Conflict Management and Peace Science*, **15**(2), 163–213.
- King, G. and Lowe, W. (2003). An automated information extraction tool for international conflict data with performance as good as human coders: A rare events evaluation design. *International Organization*, **57**(3), 617–642.
- Lang, J. and Lapata, M. (2010). Unsupervised induction of semantic roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947. Association for Computational Linguistics.
- Lehnert, W. G. (1994). Cognition, computers, and car bombs: How Yale prepared me for the 1990s. In *Beliefs, Reasoning, and Decision-Making. Psycho-Logic in Honor of Bob Abelson*, pages 143–173. Hillsdale, NJ, Hove, UK. Erlbaum. <http://ciir.cs.umass.edu/pubfiles/cognition3.pdf>.
- Li, H., Li, X., Ji, H., and Marton, Y. (2010). Domain-independent novel event discovery and semi-automatic

- event annotation. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation, Sendai, Japan, November*.
- Martin, A. D. and Quinn, K. M. (2002). Dynamic ideal point estimation via Markov chain Monte Carlo for the U.S. Supreme Court, 1953–1999. *Political Analysis*, **10**(2), 134–153.
- McClelland, C. (1970). Some effects on theory from the international event analysis movement. Mimeo, University of Southern California.
- Mimno, D., Wallach, H., and McCallum, A. (2008). Gibbs sampling for logistic normal topic models with graph-based priors. In *NIPS Workshop on Analyzing Graphs*.
- Modi, A., Titov, I., and Klementiev, A. (2012). Unsupervised induction of frame-semantic representations. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 1–7. Association for Computational Linguistics.
- Murphy, K. P. (2012). *Machine Learning: a Probabilistic Perspective*. MIT Press.
- Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, pages 705–741.
- Newman, D., Chemudugunta, C., and Smyth, P. (2006). Statistical entity-topic models. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 680–686. ACM.
- Ó Séaghdha, D. (2010). Latent variable models of selectional preference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 435–444. Association for Computational Linguistics.
- O’Brien, S. P. (2010). Crisis early warning and decision support: Contemporary approaches and thoughts on future research. *International Studies Review*, **12**(1), 87–104.
- Parker, R., Graff, D., Kong, J., Chen, K., and Maeda, K. (2009). English Gigaword Fourth Edition. *Linguistic Data Consortium*. LDC2009T13.
- Piskorski, J. and Atkinson, M. (2011). Frontex real-time news event extraction framework. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 749–752. ACM.
- Piskorski, J., Tanev, H., Atkinson, M., van der Goot, E., and Zavarella, V. (2011). Online news event extraction for global crisis surveillance. *Transactions on computational collective intelligence V*, pages 182–212.
- Polson, N. G., Scott, J. G., and Windle, J. (2012). Bayesian inference for logistic models using Polya-Gamma latent variables. *arXiv preprint arXiv:1205.0310*.
- Poon, H. and Domingos, P. (2009). Unsupervised semantic parsing. In *Proceedings of EMNLP*, pages 1–10. Association for Computational Linguistics.
- Quinn, K. M., Monroe, B. L., Colaresi, M., Crespin, M. H., and Radev, D. R. (2010). How to analyze political attention with minimal assumptions and costs. *American Journal of Political Science*, **54**(1), 209228.
- Rajaraman, A. and Ullman, J. D. (2011). Mining of massive datasets. Cambridge University Press; <http://infolab.stanford.edu/~ullman/mmds.html>.
- Ramshaw, L., Boschee, E., Freedman, M., MacBride, J., Weischedel, R., , and Zamanian, A. (2011). SERIF language processing effective trainable language understanding. *Handbook of Natural Language Processing and Machine Translation*, pages 636–644.
- Regneri, M., Koller, A., and Pinkal, M. (2010). Learning script knowledge with web experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL ’10*, pages 979–988.
- Rooth, M., Riezler, S., Prescher, D., Carroll, G., and Beil, F. (1999). Inducing a semantically annotated lexicon via EM-based clustering. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, page 104111.
- Rummel, R. (1968). The Dimensionality of Nations project.
- Rzhetsky, A., Iossifov, I., Koike, T., Krauthammer, M., Kra, P., Morris, M., Yu, H., Duboué, P. A., Weng, W., Wilbur, W. J., Hatzivassiloglou, V., and Friedman, C. (2004). GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data. *Journal of Biomedical Informatics*, **37**(1), 43–53.
- Sandhaus, E. (2008). The New York Times Annotated Corpus. *Linguistic Data Consortium*. LDC2008T19.
- Sanfilippo, A., Franklin, L., Tratz, S., Danielson, G., Mileson, N., Riensche, R., and McGrath, L. (2008). Automating frame analysis. *Social computing, behavioral modeling, and prediction*, pages 239–248.
- Schrodt, P. (2012). Precedents, progress, and prospects in political event data. *International Interactions*, **38**(4), 546–569.
- Schrodt, P. and Leetaru, K. (2013). GDELT: Global data on events, location and tone, 1979-2012. In *International Studies Association Conference*.
- Schrodt, P. A. (2001). Automated coding of international event data using sparse parsing techniques. *International Studies Association Conference*.
- Schrodt, P. A. (2006). Twenty Years of the Kansas Event Data System Project. *Political Methodologist*.
- Schrodt, P. A. and Gerner, D. J. (1994). Validity assessment of a machine-coded event data set for the Middle East, 1982-1992. *American Journal of Political Science*.
- Schrodt, P. A. and Gerner, D. J. (2004). An event data analysis of third-party mediation in the middle east and balkans. *Journal of Conflict Resolution*, **48**(3), 310–330.
- Shellman, S. M. (2004). Time series intervals and statistical inference: The effects of temporal aggregation on event data analysis. *Political Analysis*, **12**(1), 97–104.
- Titov, I. and Klementiev, A. (2011). A Bayesian model for unsupervised semantic parsing. In *Proceedings of ACL*.
- Titov, I. and Klementiev, A. (2012). A Bayesian approach to unsupervised semantic role induction. *Proceedings of EACL*.
- Wallach, H., Mimno, D., and McCallum, A. (2009). Rethinking lda: Why priors matter. *Advances in Neural Information Processing Systems*, **22**, 1973–1981.
- Yao, L., Haghghi, A., Riedel, S., and McCallum, A. (2011). Structured relation discovery using generative models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466. Association for Computational Linguistics.
- Zeger, S. L. and Karim, M. R. (1991). Generalized linear models with random effects; a Gibbs sampling approach. *Journal of the American Statistical Association*, **86**(413), 79–86.

# Graph Propagation for Paraphrasing Out-of-Vocabulary Words in Statistical Machine Translation\*

Majid Razmara<sup>1</sup>    Maryam Siahbani<sup>1</sup>    Gholamreza Haffari<sup>2</sup>    Anoop Sarkar<sup>1</sup>

<sup>1</sup> Simon Fraser University, Burnaby, BC, Canada

{razmara, msiahban, anoop}@sfu.ca

<sup>2</sup> Monash University, Clayton, VIC, Australia

reza@monash.edu

## Abstract

Out-of-vocabulary (oov) words or phrases still remain a challenge in statistical machine translation especially when a limited amount of parallel text is available for training or when there is a domain shift from training data to test data. In this paper, we propose a novel approach to finding translations for oov words. We induce a lexicon by constructing a graph on source language monolingual text and employ a graph propagation technique in order to find translations for all the source language phrases. Our method differs from previous approaches by adopting a graph propagation approach that takes into account not only one-step (from oov directly to a source language phrase that has a translation) but multi-step paraphrases from oov source language words to other source language phrases and eventually to target language translations. Experimental results show that our graph propagation method significantly improves performance over two strong baselines under intrinsic and extrinsic evaluation metrics.

## 1 Introduction

Out-of-vocabulary (oov) words or phrases still remain a challenge in statistical machine translation. SMT systems usually copy unknown words verbatim to the target language output. Although this is helpful in translating a small fraction of oovs such as named entities for languages with same writing systems, it harms the translation in other types of oovs and distant language pairs. In general, copied-over oovs are a hindrance to fluent, high quality translation, and we can see evidence of this in automatic measures such as BLEU (Papineni et al., 2002) and also in human evaluation scores such as HTER. The problem becomes more severe when only a limited amount of parallel text is available for training or when the training and test data are from different domains. Even noisy translation of oovs can aid the language model to better

re-order the words in the target language (Zhang et al., 2012).

Increasing the size of the parallel data can reduce the number of oovs. However, there will always be some words or phrases that are new to the system and finding ways to translate such words or phrases will be beneficial to the system. Researchers have applied a number of approaches to tackle this problem. Some approaches use pivot languages (Callison-Burch et al., 2006) while others use lexicon-induction-based approaches from source language monolingual corpora (Koehn and Knight, 2002; Garera et al., 2009; Marton et al., 2009).

Pivot language techniques tackle this problem by taking advantage of available parallel data between the source language and a third language. Using a pivot language, oovs are translated into a third language and back into the source language and thereby paraphrases to those oov words are extracted (Callison-Burch et al., 2006). For each oov, the system can be augmented by aggregating the translations of all its paraphrases and assign them to the oov. However, these methods require parallel corpora between the source language and one or multiple pivot languages.

Another line of work exploits spelling and morphological variants of oov words. Habash (2008) presents techniques for online handling of oov words for Arabic to English such as spelling expansion and morphological expansion. Huang et al. (2011) proposes a method to combine sublexical/constituent translations of an oov word or phrase to generate its translations.

Several researchers have applied lexicon-induction methods to create a bilingual lexicon for those oovs. Marton et al. (2009) use a monolingual text on the source side to find paraphrases to oov words for which the translations are available. The translations for these paraphrases are

\*This research was partially supported by an NSERC, Canada (RGPIN: 264905) grant. The third author was supported by an early career research award from Monash University to visit Simon Fraser University.

then used as the translations of the oov word. These methods are based on the *distributional hypothesis* which states that words appearing in the same contexts tend to have similar meaning (Harris, 1954). Marton et al. (2009) showed that this method improves over the baseline system where oovs are untranslated.

We propose a graph propagation-based extension to the approach of Marton et al. (2009) in which a graph is constructed from source language monolingual text<sup>1</sup> and the source-side of the available parallel data. Nodes that have related meanings are connected together and nodes for which we have translations in the phrase-table are annotated with target-side translations and their feature values. A graph propagation algorithm is then used to propagate translations from labeled nodes to unlabeled nodes (phrases appearing only in the monolingual text and oovs). This provides a general purpose approach to handle several types of oovs, including morphological variants, spelling variants and synonyms<sup>2</sup>.

Constructing such a huge graph and propagating messages through it pose severe computational challenges. Throughout the paper, we will see how these challenges are dealt with using scalable algorithms.

## 2 Collocational Lexicon Induction

Rapp (1995) introduced the notion of a distributional profile in bilingual lexicon induction from monolingual data. A *distributional profile* (DP) of a word or phrase type is a co-occurrence vector created by combining all co-occurrence vectors of the tokens of that phrase type. Each distributional profile can be seen as a point in a  $|V|$ -dimensional space where  $V$  is the vocabulary where each word type represents a unique axis. Points (i.e. phrase types) that are close to one another in this high-dimensional space can represent paraphrases. This approach has also been used in machine translation to find in-vocabulary paraphrases for oov words on the source side and find a way to translate them.

### 2.1 Baseline System

Marton et al. (2009) was the first to successfully integrate a collocational approach to finding trans-

<sup>1</sup>Here on by monolingual data we always mean monolingual data on the source language

<sup>2</sup>Named entity oovs may be handled properly by copying or transliteration.

lations for oov words into an end-to-end SMT system. We explain their method in detail as we will compare against this approach. The method relies on monolingual distributional profiles (DPs) which are numerical vectors representing the context around each word. The goal is to find words or phrases that appear in similar contexts as the oovs. For each oov a distributional profile is created by collecting all words appearing in a fixed distance from all occurrences of the oov word in the monolingual text. These co-occurrence counts are converted to an association measure (Section 2.2) that encodes the relatedness of each pair of words or phrases.

Then, the most similar phrases to each oov are found by measuring the similarity of their DPs to that of the oov word. Marton et al. (2009) uses a heuristic to prune the search space for finding candidate paraphrases by keeping the surrounding context (e.g.  $L\_R$ ) of each occurrences of the oov word. All phrases that appear in any of such contexts are collected as candidate paraphrases. For each of these paraphrases, a DP is constructed and compared to that of the oov word using a similarity measure (Section 2.2).

The top-k paraphrases that have translations in the phrase-table are used to assign translations and scores to each oov word by marginalizing translations over paraphrases:

$$p(t|o) = \sum_s p(t|s)p(s|o)$$

where  $t$  is a phrase on the target side,  $o$  is the oov word or phrase, and  $s$  is a paraphrase of  $o$ .  $p(s|o)$  is estimated using a similarity measure over DPs and  $p(t|s)$  is coming from the phrase-table.

We reimplemented this collocational approach for finding translations for oovs and used it as a baseline system.

Alternative ways of modeling and comparing distributional profiles have been proposed (Rapp, 1999; Fung and Yee, 1998; Terra and Clarke, 2003; Garera et al., 2009; Marton et al., 2009). We review some of them here and compare their performance in Section 4.3.

### 2.2 Association Measures

Given a word  $u$ , its distributional profile  $DP(u)$  is constructed by counting surrounding words (in a fixed window size) in a monolingual corpus.

$$DP(u) = \{\langle A(u, w_i) \rangle \mid w_i \in V\}$$

The counts can be collected in positional<sup>3</sup> (Rapp, 1999) or non-positional way (count all the word occurrences within the sliding window).  $A(\cdot, \cdot)$  is an association measure and can simply be defined as co-occurrence counts within sliding windows. Stronger association measures can also be used such as:

**Conditional probability:** the probability for the occurrence of each word in DP given the occurrence of  $u$ :  $CP(u, w_i) = P(w_i|u)$  (Schütze and Pedersen, 1997)

**Pointwise Mutual Information:** this measure is a transformation of the independence assumption into a ratio. Positive values indicate that words co-occur more than what we expect under the independence assumption (Lin, 1998):

$$PMI(u, w_i) = \log_2 \frac{P(u, w_i)}{P(u)P(w_i)}$$

**Likelihood ratio:** (Dunning, 1993) uses the likelihood ratio for word similarity:

$$\lambda(u, w_i) = \frac{L(P(w_i|u); p) * L(P(w_i|\neg u); p)}{L(P(w_i|u); p_1) * L(P(w_i|\neg u); p_2)}$$

where  $L$  is likelihood function under the assumption that word counts in text have binomial distributions. The numerator represents the likelihood of the hypothesis that  $u$  and  $w_i$  are independent ( $P(w_i|u) = P(w_i|\neg u) = p$ ) and the denominator represents the likelihood of the hypothesis that  $u$  and  $w_i$  are dependent ( $P(w_i|u) \neq P(w_i|\neg u)$ ,  $P(w_i|u) = p_1$ ,  $P(w_i|\neg u) = p_2$ )<sup>4</sup>.

**Chi-square test:** is a statistical hypothesis testing method to evaluate independence of two categorical random variables, e.g. whether the *occurrence* of  $u$  and  $w_i$  (denoted by  $x$  and  $y$  respectively) are independent. The test statistics  $\chi^2(u, w_i)$  is the deviation of the observed counts  $f_{x,y}$  from their expected values  $E_{x,y}$ :

$$\chi^2(u, w_i) := \sum_{x \in \{w_i, \neg w_i\}} \sum_{y \in \{u, \neg u\}} \frac{(f_{x,y} - E_{x,y})^2}{E_{x,y}}$$

### 2.3 Similarity Measures

Various functions have been used to estimate the similarity between distributional profiles.

<sup>3</sup>e.g., position 1 is the word immediately after, position -1 is the word immediately before etc.

<sup>4</sup>Binomial distribution  $B(k; n, \theta)$  gives the probability of observing  $k$  heads in  $n$  tosses of a coin where the coin parameter is  $\theta$ . In our context,  $p$ ,  $p_1$  and  $p_2$  are parameters of Binomial distributions estimated using maximum likelihood.

Given two distributional profiles  $DP(u)$  and  $DP(v)$ , some similarity functions can be defined as follows. Note that  $A(\cdot, \cdot)$  stands for the various association measures defined in Sec. 2.2.

**Cosine coefficient** is the cosine the angle between two vectors  $DP(u)$  and  $DP(v)$ :

$$\cos(DP(u), DP(v)) = \frac{\sum_{w_i \in V} A(u, w_i)A(v, w_i)}{\sqrt{\sum_{w_i \in V} A(u, w_i)^2} \sqrt{\sum_{w_i \in V} A(v, w_i)^2}}$$

**$L_1$ -Norm** computes the accumulated distance between entries of two distributional profiles ( $L_1(\cdot, \cdot)$ ). It has been used as word similarity measure in language modeling (Dagan et al., 1999).

$$L_1(DP(u), DP(v)) = \sum_{w_i \in V} |A(u, w_i) - A(v, w_i)|$$

**Jensen-Shannon Divergence** is a symmetric version of *contextual average mutual information* ( $KL$ ) which is used by (Dagan et al., 1999) as word similarity measure.

$$JSD(DP(u), DP(v)) = KL(DP(u), AVG_{DP}(u, v)) + KL(DP(v), AVG_{DP}(u, v))$$

$$AVG_{DP}(u, v) = \left\{ \frac{A(u, w_i) + A(v, w_i)}{2} \mid w_i \in V \right\}$$

$$KL(DP(u), DP(v)) = \sum_{w_i \in V} A(u, w_i) \log \frac{A(u, w_i)}{A(v, w_i)}$$

## 3 Graph-based Lexicon Induction

We propose a novel approach to alleviate the oov problem. Given a (possibly small amount of) parallel data between the source and target languages, and a large monolingual data in the source language, we construct a graph over all phrase types in the monolingual text and the source side of the parallel corpus and connect phrases that have similar meanings (i.e. appear in similar context) to one another. To do so, the distributional profiles of all source phrase types are created. Each phrase type represents a vertex in the graph and is connected to other vertices with a weight defined by a similarity measure between the two profiles (Section 2.3). There are three types of vertices in the graph: i) labeled nodes which appear in the parallel corpus and for which we have the target-side

translations<sup>5</sup>; ii) oov nodes from the *dev/test set* for which we seek labels (translations); and iii) unlabeled nodes (words or phrases) from the *monolingual data* which appear usually between oov nodes and labeled nodes. When a relatively small parallel data is used, unlabeled nodes outnumber labeled ones and many of them lie on the paths between an oov node to labeled ones.

Marton et al. (2009)’s approach ignores these bridging nodes and connects each oov node to the  $k$ -nearest *labeled* nodes. One may argue that these unlabeled nodes do not play a major role in the graph and the labels will eventually get to the oov nodes from the labeled nodes by directly connecting them. However based on the definition of the similarity measures using context, it is quite possible that an oov node and a labeled node which are connected to the same unlabeled node do not share any context words and hence are not directly connected. For instance, consider three nodes,  $u$  (unlabeled),  $o$  (oov) and  $l$  (labeled) where  $u$  has the same left context words with  $o$  but share the right context with  $l$ .  $o$  and  $l$  are not connected since they do not share any context word.

Once a graph is constructed based on similarities of phrases, graph propagation is used to propagate the labels from labeled nodes to unlabeled and oov nodes. The approach is based on the *smoothness assumption* (Chapelle et al., 2006) which states if two nodes are similar according to the graph, then their output labels should also be similar.

The baseline approach (Marton et al., 2009) can be formulated as a *bipartite graph* with two types of nodes: labeled nodes ( $L$ ) and oov nodes ( $O$ ). Each oov node is connected to a number of labeled nodes, and vice versa and there is no edge between nodes of the same type. In such a graph, the similarity of each pair of nodes is computed using one of the similarity measures discussed above. The labels are translations and their probabilities (more specifically  $p(e|f)$ ) from the phrase-table extracted from the parallel corpus. Translations get propagated to oov nodes using a label propagation technique. However beside the difference in the oov label assignment, there is a major difference between our bipartite graph and the baseline (Marton et al., 2009): we do not use a heuristic to

<sup>5</sup>It is possible that a phrase appears in the parallel corpus, but not in the phrase-table. This happens when the word-alignment module is not able to align the phrase to a target side word or words.

reduce the number of neighbor candidates and we consider all possible candidates that share at least one context word. This makes a significant difference in practice as shown in Section 4.3.1.

We also take advantage of unlabeled nodes to help connect oov nodes to labeled ones. The discussed bipartite graph can easily be expanded to a *tripartite graph* by adding unlabeled nodes. Figure 1 illustrate a tripartite graph in which unlabeled nodes are connected to both labeled and oov nodes. Again, there is no edge between nodes of the same type. We also created the *full graph* where all nodes can be freely connected to nodes of any type including the same type. However, constructing such graph and doing graph propagation on it is computationally very expensive for large  $n$ -grams.

### 3.1 Label Propagation

Let  $G = (V, E, W)$  be a graph where  $V$  is the set of vertices,  $E$  is the set of edges, and  $W$  is the edge weight matrix. The vertex set  $V$  consists of labeled  $V_L$  and unlabeled  $V_U$  nodes, and the goal of the labeling propagation algorithm is to compute soft labels for unlabeled vertices from the labeled vertices. Intuitively, the edge weight  $W(u, v)$  encodes the degree of our belief about the similarity of the soft labeling for nodes  $u$  and  $v$ . A soft label  $\hat{Y}_v \in \Delta^{m+1}$  is a probability vector in  $(m + 1)$ -dimensional simplex, where  $m$  is the number of possible labels and the additional dimension accounts for the *undefined*  $\perp$  label<sup>6</sup>.

In this paper, we make use of the *modified Adsorption* (MAD) algorithm (Talukdar and Crammer, 2009) which finds soft label vectors  $\hat{Y}_v$  to solve the following unconstrained optimization problem:

$$\min_{\hat{Y}} \mu_1 \sum_{v \in V_L} p_{1,v} \|Y_v - \hat{Y}_v\|_2^2 + \quad (1)$$

$$\mu_2 \sum_{v,u} p_{2,v} W_{v,u} \|\hat{Y}_v - \hat{Y}_u\|_2^2 + \quad (2)$$

$$\mu_3 \sum_v p_{3,v} \|\hat{Y}_v - R_v\|_2^2 \quad (3)$$

where  $\mu_i$  and  $p_{i,v}$  are hyper-parameters ( $\forall v : \sum_i p_{i,v} = 1$ )<sup>7</sup>, and  $R_v \in \Delta^{m+1}$  encodes our prior belief about the labeling of a node  $v$ . The first

<sup>6</sup>Capturing those cases where the given data is not enough to reliably compute a soft labeling using the initial  $m$  real labels.

<sup>7</sup>The values of these hyper-parameters are set to their defaults in the *Junto* toolkit (Talukdar and Crammer, 2009).

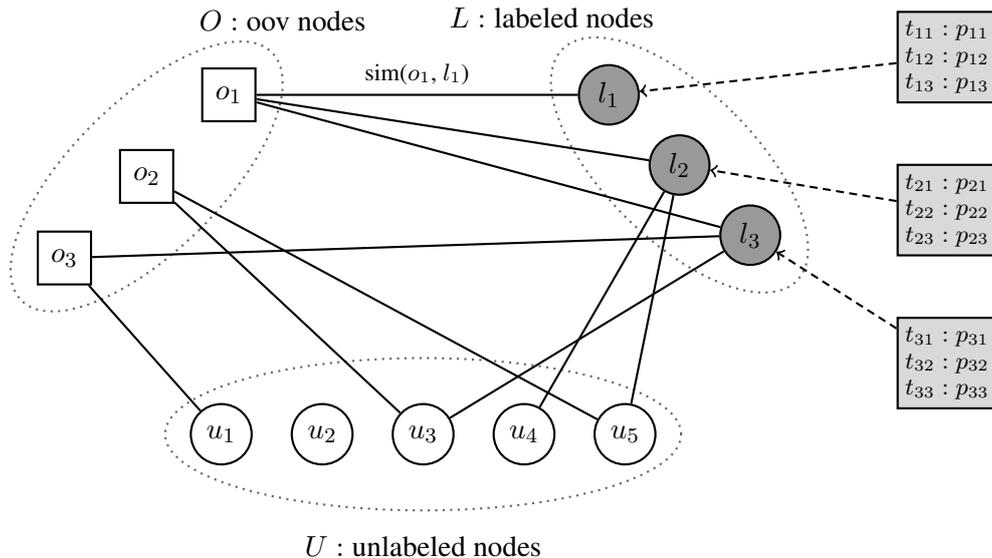


Figure 1: A tripartite graph between oov, labeled and unlabeled nodes. Translations propagate either directly from labeled nodes to oov nodes or indirectly via unlabeled nodes.

term (1) enforces the labeling of the algorithm to match the seed labeling  $Y_v$  with different extent for different labeled nodes. The second term (2) enforces the *smoothness* of the labeling according to the graph structure and edge weights. The last term (3) regularizes the soft labeling for a vertex  $v$  to match a priori label  $R_v$ , e.g. for high-degree unlabeled nodes (hubs in the graph) we may believe that the neighbors are not going to produce reliable label and hence the probability of undefined label  $\perp$  should be higher. The optimization problem can be solved with an efficient iterative algorithm which is parallelized in a MapReduce framework (Talukdar et al., 2008; Rao and Yarowsky, 2009). We used the *Junto label propagation* toolkit (Talukdar and Crammer, 2009) for label propagation.

### 3.2 Efficient Graph Construction

Graph-based approaches can easily become computationally very expensive as the number of nodes grow. In our case, we use phrases in the monolingual text as graph vertices. These phrases are n-grams up to a certain value, which can result in millions of nodes. For each node a distributional profile (DP) needs to be created. The number of possible edges can easily explode in size as there can be as many as  $O(n^2)$  edges where  $n$  is the number of nodes. A common practice to control the number of edges is to connect each node to at most  $k$  other nodes (k-nearest neigh-

bor). However, finding the top-k nearest nodes to each node requires considering its similarity to all the other nodes which requires  $O(n^2)$  computations and since  $n$  is usually very large, doing such is practically intractable. Therefore, researchers usually resort to an approximate k-NN algorithms such as *locality-sensitive hashing* (?; Goyal et al., 2012).

Fortunately, since we use context words as cues for relating their meaning and since the similarity measures are defined based on these cues, the number of neighbors we need to consider for each node is reduced by several orders of magnitude. We incorporate an inverted-index-style data structure which indicates what nodes are neighbors based on each context word. Therefore, the set of neighbors of a node consists of union of all the neighbors bridged by each context word in the DP of the node. However, the number of neighbors to be considered for each node even after this drastic reduction is still large (in order of a few thousands).

In order to deal with the computational challenges of such a large graph, we take advantage of the Hadoop’s MapReduce functionality to do both graph construction and label propagation steps.

## 4 Experiments & Results

### 4.1 Experimental Setup

We experimented with two different domains for the bilingual data: *Europarl* corpus (v7) (Koehn,

Dataset	Domain	Sents	Tokens	
			Fr	En
Bitext	Europarl	10K	298K	268K
	EMEA	1M	16M	14M
Monotext	Europarl	2M	60M	–
Dev-set	WMT05	2K	67K	58K
Test-set	WMT05	2K	66K	58K

Table 1: Statistics of training sets in different domains.

2005), and *European Medicines Agency* documents (EMEA) (Tiedemann, 2009) from French to English. For the monolingual data, we used French side of the Europarl corpus and we used ACL/WMT 2005<sup>8</sup> data for dev/test sets. Table 1 summarizes statistics of the datasets used.

From the dev and test sets, we extract all source words that do not appear in the phrase-table constructed from the parallel data. From the oovs, we exclude numbers as well as named entities. We apply a simple heuristic to detect named entities: basically words that are capitalized in the original dev/test set that do not appear at the beginning of a sentence are named entities. Table 2 shows the number of oov types and tokens for Europarl and EMEA systems in both dev and test sets.

Dataset	Dev		Test	
	types	tokens	types	tokens
Europarl	1893	2229	1830	2163
EMEA	2325	4317	2294	4190

Table 2: number of oovs in dev and test sets for Europarl and EMEA systems.

For the end-to-end MT pipeline, we used Moses (Koehn et al., 2007) with these standard features: relative-frequency and lexical translation model (TM) probabilities in both directions; distortion model; language model (LM) and word count. Word alignment is done using GIZA++ (Och and Ney, 2003). We used distortion limit of 6 and max-phrase-length of 10 in all the experiments. For the language model, we used the KenLM toolkit (Heafield, 2011) to create a 5-gram language model on the target side of the Europarl corpus (v7) with approximately 54M tokens with Kneser-Ney smoothing.

#### 4.1.1 Phrase-table Integration

Once the translations and their probabilities for each oov are extracted, they are added to the

<sup>8</sup><http://www.statmt.org/wpt05/mt-shared-task/>

phrase-table that is induced from the parallel text. The probability for new entries are added as a new feature in the log-linear framework to be tuned along with other features. The value of this newly introduced feature for original entries in the phrase-table is set to 1. Similarly, the value of original four probability features in the phrase-table for the new entries are set to 1. The entire training pipeline is as follows: (i) a phrase table is constructed using parallel data as usual, (ii) oovs for dev and test sets are extracted, (iii) oovs are translated using graph propagation, (iv) oovs and translations are added to the phrase table, introducing a new feature type, (v) the new phrase table is tuned (with a LM) using MERT (Och, 2003) on the dev set.

## 4.2 Evaluation

If we have a list of possible translations for oovs with their probabilities, we become able to evaluate different methods we discussed. We word-aligned the dev/test sets by concatenating them to a large parallel corpus and running GIZA++ on the whole set. The resulting word alignments are used to extract the translations for each oov. The correctness of this gold standard is limited to the size of the parallel data used as well as the quality of the word alignment software toolkit, and is not 100% precise. However, it gives a good estimate of how each oov should be translated without the need for human judgments.

For evaluating our baseline as well as graph-based approaches, we use both intrinsic and extrinsic evaluations. Two intrinsic evaluation metrics that we use to evaluate the possible translations for oovs are *Mean Reciprocal Rank* (MRR) (Voorhees, 1999) and *Recall*. Intrinsic evaluation metrics are faster to apply and are used to optimize different hyper-parameters of the approach (e.g. window size, phrase length, etc.). Once we come up with the optimized values for the hyper-parameters, we extrinsically evaluate different approaches by adding the new translations to the phrase-table and run it through the MT pipeline.

### 4.2.1 MRR

MRR is an Information Retrieval metric used to evaluate any process that produces a ranked list of possible candidates. The reciprocal rank of a list is the inverse of the rank of the correct answer in the list. Such score is averaged over a set, oov set

in our case, to get the mean-reciprocal-rank score.

$$\text{MRR} = \frac{1}{|O|} \sum_{i=1}^{|O|} \frac{1}{\text{rank}_i} \quad O = \{oov\}$$

In a few cases, there are multiple translations for an oov word (i.e. appearing more than once in the parallel corpus and being assigned to multiple different phrases), we take the average of reciprocal ranks for each of them.

#### 4.2.2 Recall

MRR takes the probabilities of oov translations into account in sorting the list of candidate translations. However, in an MT pipeline, the language model is supposed to rerank the hypotheses and move more appropriate translations (in terms of fluency) to the top of the list. Hence, we also evaluate our candidate translation regardless of the ranks. Since Moses uses a certain number of translations per source phrase (called the translation table limit or *ttl* which we set to 20 in our experiments), we use the *recall* measure to evaluate the top *ttl* translations in the list. Recall is another Information Retrieval measure that is the fraction of correct answers that are retrieved. For example, it assigns score of 1 if the correct translation of the oov word is in the top-k list and 0 otherwise. The scores are averaged over all oovs to compute recall.

$$\text{Recall} = \frac{|\{\text{gold standard}\} \cap \{\text{candidate list}\}|}{|\{\text{gold standard}\}|}$$

#### 4.3 Intrinsic Results

In Section 2.2 and 2.3, different types of association measures and similarity measures have been explained to build and compare distributional profiles. Table 3 shows the results on Europarl when using different similarity combinations. The measures are evaluated by fixing the window size to 4 and maximum candidate paraphrase length to 2 (e.g. bigram). First column shows the association measures used to build DPs. As the results show, the combination of PMI as association measure and cosine as DP similarity measure outperforms the other possible combinations. We use these two measures throughout the rest of the experiments.

Figure 2 illustrates the effects of different window sizes and paraphrase lengths on MRR. As the figure shows, the best MRR is reached when using window size of 4 and trigram nodes. Going from trigram to 4-gram results in a drop in MRR. One

Assoc	cosine(%)		$L_1$ norm(%)		JSD(%)	
	MRR	RCL	MRR	RCL	MRR	RCL
CP	1.66	4.16	2.18	5.55	2.33	6.32
LLR	1.79	4.26	0.13	0.37	0.5	1.00
PMI	<b>3.91</b>	<b>7.75</b>	0.50	1.17	0.59	1.21
Chi	1.66	4.16	0.26	0.55	0.03	0.05

Table 3: Results of intrinsic evaluations (MRR and Recall) on Europarl, window size 4 and paraphrase length 2

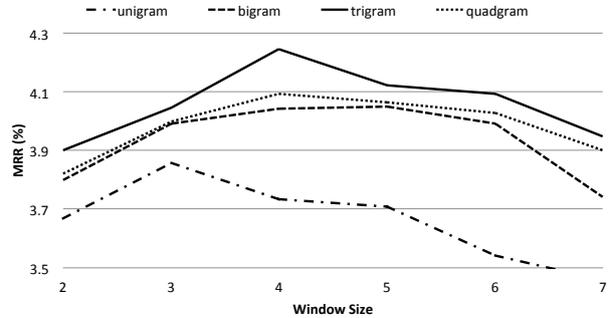


Figure 2: Effects of different window sizes and paraphrase length on the MRR of the dev set.

reason would be that distributional profiles for 4-grams are very sparse and that negatively affects the stability of similarity measures.

Figure 3 illustrates the effect of increasing the size of monolingual text on both MRR and recall.  $1\times$  refers to the case of using  $125k$  sentences for the monolingual text and the  $16\times$  indicates using the whole Europarl text on the source side ( $\approx 2M$  sentences). As shown, there is a linear correlation between the logarithm of the data size and the MRR and recall ratios. Interestingly, MRR is growing faster than recall by increasing the monolingual text size, which means that the scoring function gets better when more data is available. The figure also indicates that a much bigger monolingual text data can be used to further improve the quality of the translations, however, at the expense of more computational resources.

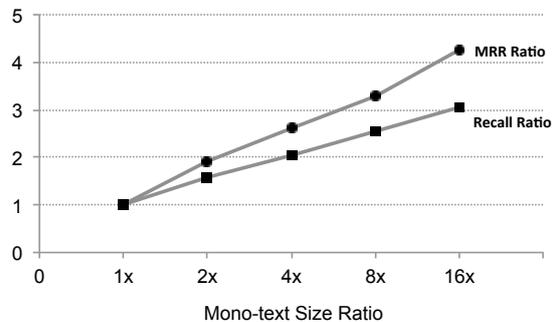


Figure 3: Effect of increasing the monolingual text size on MRR and Recall.

Graph	Neighbor	MRR %	RCL %
Bipartite	20	5.2	12.5
Tripartite	15+5	5.9	12.6
Full	20	5.1	10.9
Baseline	20	3.7	7.2

Table 4: Intrinsic results of different types of graphs when using unigram nodes on Europarl.

Type	Node	MRR %	RCL %
Bipartite	unigram	5.2	12.5
	bigram	6.8	15.7
Tripartite	unigram	5.9	12.6
	bigram	6.9	15.9
Baseline	bigram	3.9	7.7

Table 5: Results on using unigram or bigram nodes.

### 4.3.1 Graph-based Results

Table 4 shows the intrinsic results on the Europarl corpus when using unigram nodes in each of the graphs. The results are evaluated on the dev-set based on the gold alignment created using GIZA++. Each node is connected to at most 20 other nodes (same as the max-paraphrase-limit in the baseline). For the tripartite graph, each node is connected to 15 labeled nodes and 5 unlabeled ones. The tripartite graph gets a slight improvement over the bipartite one, however, the full graph failed to have the same increase. One reason is that allowing paths longer than 2 between oov and labeled nodes causes more noise to propagate into the graph. In other words, a paraphrase of a paraphrase of a paraphrase is not necessarily a useful paraphrase for an oov as the translation may no longer be a valid one.

Table 5 also shows the effect of using bigrams instead of unigrams as graph nodes. There is an improvement by going from unigrams to bigrams in both bipartite and tripartite graphs. We did not use trigrams or larger n-grams in our experiments.

### 4.4 Extrinsic Results

The generated candidate translations for the oovs can be added to the phrase-table created using the parallel corpus to increase the coverage of the phrase-table. This aggregated phrase-table is to be tuned along with the language model on the dev set, and run on the test set. BLEU (Papineni et al., 2002) is still the de facto evaluation metric for machine translation and we use that to measure the quality of our proposed approaches for MT.

In these experiments, we do not use alignment information on dev or test sets unlike the previous section.

Table 6 reports the Bleu scores for different domains when the oov translations from the graph propagation is added to the phrase-table and compares them with the baseline system (i.e. Moses). Results for our approach is based on unigram tripartite graphs and show that we improve over the baseline in both the same-domain (Europarl) and domain adaptation (EMEA) settings.

Table 7 shows some translations found by our system for oov words.

oov	gold standard	candidate list
spécialement	undone particularly especially special particular	particularly specific only particular should and especially
assentiment	approval	support agreement approval accession will approve endorses

Table 7: Two examples of oov translations found by our method.

## 5 Related work

There has been a long line of research on learning translation pairs from non-parallel corpora (Rapp, 1995; Koehn and Knight, 2002; Haghghi et al., 2008; Garera et al., 2009; Marton et al., 2009; Laws et al., 2010). Most have focused on extracting a translation lexicon by mining monolingual resources of data to find clues, using probabilistic methods to map words, or by exploiting the cross-language evidence of closely related languages. Most of them evaluated only high-frequency words of specific types (nouns or content words) (Rapp, 1995; Koehn and Knight, 2002; Haghghi et al., 2008; Garera et al., 2009; Laws et al., 2010) In contrast, we do not consider any constraint on our test data and our data includes many low frequency words. It has been shown that translation of high-frequency words is easier than low frequency words (Tamura et al., 2012).

Some methods have used a third language(s) as pivot or bridge to find translation pairs (Mann and Yarowsky, 2001; Schafer and Yarowsky, 2002; Callison-Burch et al., 2006).

Corpus	System	MRR	Recall	Dev Bleu	Test Bleu
Europarl	Baseline	–	–	28.53	28.97
	Our approach	5.9	12.6	28.76	29.40*
EMEA	Baseline	–	–	20.05	20.34
	Our approach	3.6	7.4	20.54	20.80*

\* Statistically significant with  $p < 0.02$  using the bootstrap resampling significance test (in Moses).

Table 6: Bleu scores for different domains with or without using oov translations.

Context similarity has been used effectively in bilingual lexicon induction (Rapp, 1995; Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009; Marton et al., 2009; Laws et al., 2010). It has been modeled in different ways: in terms of adjacent words (Rapp, 1999; Fung and Yee, 1998), or dependency relations (Garera et al., 2009). Laws et al. (2010) used linguistic analysis in the form of graph-based models instead of a vector space. But all of these researches used an available seed lexicon as the basic source of similarity between source and target languages unlike our method which just needs a monolingual corpus of source language which is freely available for many languages and a small bilingual corpora.

Some methods tried to alleviate the lack of seed lexicon by using orthographic similarity to extract a seed lexicon (Koehn and Knight, 2002; Fiser and Ljubesic, 2011). But it is not a practical solution in case of unrelated languages.

Haghighi et al. (2008) and Daumé and Jagarlamudi (2011) proposed generative models based on canonical correlation analysis to extract translation lexicons for non-parallel corpora by learning a matching between source and target lexicons. Using monolingual features to represent words, feature vectors are projected from source and target words into a canonical space to find the appropriate matching between them. Their method relies on context features which need a seed lexicon and orthographic features which only works for phylogenetically related languages.

Graph-based semi-supervised methods have been shown to be useful for domain adaptation in MT as well. Alexandrescu and Kirchhoff (2009) applied a graph-based method to determine similarities between sentences and use these similarities to promote similar translations for similar sentences. They used a graph-based semi-supervised model to re-rank the n-best translation hypothesis. Liu et al. (2012) extended Alexandrescu’s model to use translation consensus among simi-

lar sentences in bilingual training data by developing a new structured label propagation method. They derived some features to use during decoding process that has been shown useful in improving translation quality. Our graph propagation method connects monolingual source phrases with oovs to obtain translation and so is a very different use of graph propagation from these previous works.

Recently label propagation has been used for lexicon induction (Tamura et al., 2012). They used a graph based on context similarity as well as co-occurrence graph in propagation process. Similar to our approach they used unlabeled nodes in label propagation process. However, they use a seed lexicon to define labels and comparable corpora to construct graphs unlike our approach.

## 6 Conclusion

We presented a novel approach for inducing oov translations from a monolingual corpus on the source side and a parallel data using graph propagation. Our results showed improvement over the baselines both in intrinsic evaluations and on BLEU. Future work includes studying the effect of size of parallel corpus on the induced oov translations. Increasing the size of parallel corpus on one hand reduces the number of oovs. But, on the other hand, there will be more labeled paraphrases that increases the chance of finding the correct translation for oovs in the test set.

Currently, we find paraphrases for oov words. However, oovs can be considered as n-grams (phrases) instead of unigrams. In this scenario, we also can look for paraphrases and translations for phrases containing oovs and add them to the phrase-table as new translations along with the translations for unigram oovs.

We also plan to explore different graph propagation objective functions. Regularizing these objective functions appropriately might let us scale to much larger data sets with an order of magnitude more nodes in the graph.

## References

- Andrei Alexandrescu and Katrin Kirchhoff. 2009. Graph-based learning for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 119–127, Stroudsburg, PA, USA. Association for Computational Linguistics.
- C. Callison-Burch, P. Koehn, and M. Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24. Association for Computational Linguistics.
- O. Chapelle, B. Schölkopf, and A. Zien, editors. 2006. *Semi-Supervised Learning*. MIT Press, Cambridge, MA.
- Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Mach. Learn.*, 34(1-3):43–69, February.
- Hal Daumé, III and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 407–412, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74, March.
- Darja Fiser and Nikola Ljubesic. 2011. Bilingual lexicon extraction from comparable corpora for closely related languages. In *RANLP*, pages 125–131.
- Pascale Fung and Lo Yuen Yee. 1998. An ir approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 414–420. Association for Computational Linguistics.
- Nikesh Garera, Chris Callison-Burch, and David Yarowsky. 2009. Improving translation lexicon induction from monolingual corpora via dependency contexts and part-of-speech equivalences. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 129–137, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Amit Goyal, Hal Daume III, and Raul Guerra. 2012. Fast Large-Scale Approximate Graph Construction for NLP. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '12.
- Nizar Habash. 2008. Four techniques for online handling of out-of-vocabulary words in arabic-english statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 57–60. Association for Computational Linguistics.
- Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *ACL*, pages 771–779.
- Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.
- Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197.
- Chung-Chi Huang, Ho-Ching Yen, Ping-Che Yang, Shih-Ting Huang, and Jason S Chang. 2011. Using sublexical translations to handle the oov problem in machine translation. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(3):16.
- Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition - Volume 9*, ULA '02, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. ACL.
- P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5.
- Florian Laws, Lukas Michelbacher, Beate Dorow, Christian Scheible, Ulrich Heid, and Hinrich Schütze. 2010. A linguistically grounded graph model for bilingual lexicon extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 614–622, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2*, ACL '98, pages 768–774, Stroudsburg, PA, USA. Association for Computational Linguistics.

- Shujie Liu, Chi-Ho Li, Mu Li, and Ming Zhou. 2012. Learning translation consensus with structured label propagation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 302–310, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Gideon S. Mann and David Yarowsky. 2001. Multipath translation lexicon induction via bridge languages. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA.
- Yuval Marton, Chris Callison-Burch, and Philip Resnik. 2009. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 381–390, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.
- Franz Josef Och. 2003. Minimum error rate training for statistical machine translation. In *Proceedings of the 41th Annual Meeting of the ACL*, Sapporo, July. ACL.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Delip Rao and David Yarowsky. 2009. Ranking and semi-supervised classification on large scale graphs using map-reduce. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, TextGraphs-4. Association for Computational Linguistics.
- Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 320–322. Association for Computational Linguistics.
- Reinhard Rapp. 1999. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 519–526. Association for Computational Linguistics.
- Charles Schafer and David Yarowsky. 2002. Inducing translation lexicons via diverse similarity measures and bridge languages. In *proceedings of the 6th conference on Natural language learning - Volume 20*, COLING-02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hinrich Schütze and Jan O. Pedersen. 1997. A cooccurrence-based thesaurus and two applications to information retrieval. *Inf. Process. Manage.*, 33(3):307–318, May.
- Partha Pratim Talukdar and Koby Crammer. 2009. New Regularized Algorithms for Transductive Learning. In *European Conference on Machine Learning (ECML-PKDD)*.
- Partha Pratim Talukdar, Joseph Reisinger, Marius Paşca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08.
- Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2012. Bilingual lexicon extraction from comparable corpora using label propagation. In *EMNLP-CoNLL*, pages 24–36.
- Egídio L. Terra and Charles L. A. Clarke. 2003. Frequency estimates for statistical word similarity measures. In *HLT-NAACL*.
- Jörg Tiedemann. 2009. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia.
- Ellen M. Voorhees. 1999. TREC-8 Question Answering Track Report. In *Proceedings of the 8th Text Retrieval Conference*, pages 77–82.
- Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2012. Handling unknown words in statistical machine translation from a new perspective. In *Natural Language Processing and Chinese Computing*, pages 176–187. Springer.

# Online Relative Margin Maximization for Statistical Machine Translation

**Vladimir Eidelman**  
Computer Science  
and UMIACS  
University of Maryland  
College Park, MD  
vlad@umiacs.umd.edu

**Yuval Marton**  
Microsoft  
City Center Plaza  
Bellevue, WA  
yuvalmarton@gmail.com

**Philip Resnik**  
Linguistics  
and UMIACS  
University of Maryland  
College Park, MD  
resnik@umd.edu

## Abstract

Recent advances in large-margin learning have shown that better generalization can be achieved by incorporating higher order information into the optimization, such as the spread of the data. However, these solutions are impractical in complex structured prediction problems such as statistical machine translation. We present an online gradient-based algorithm for relative margin maximization, which bounds the spread of the projected data while maximizing the margin. We evaluate our optimizer on Chinese-English and Arabic-English translation tasks, each with small and large feature sets, and show that our learner is able to achieve significant improvements of 1.2-2 BLEU and 1.7-4.3 TER on average over state-of-the-art optimizers with the large feature set.

## 1 Introduction

The desire to incorporate high-dimensional sparse feature representations into statistical machine translation (SMT) models has driven recent research away from Minimum Error Rate Training (MERT) (Och, 2003), and toward other discriminative methods that can optimize more features. Examples include minimum risk (Smith and Eisner, 2006), pairwise ranking (PRO) (Hopkins and May, 2011), RAMPION (Gimpel and Smith, 2012), and variations of the margin-infused relaxation algorithm (MIRA) (Watanabe et al., 2007; Chiang et al., 2008; Cherry and Foster, 2012). While the objective function and optimization method vary for each optimizer, they can all be broadly described as learning a linear model, or parameter vector  $\mathbf{w}$ , which is used to score alternative translation hypotheses.

In every SMT system, and in machine learning in general, the goal of learning is to find a

model that generalizes well, i.e. one that will yield good translations for previously unseen sentences. However, as the dimension of the feature space increases, generalization becomes increasingly difficult. Since only a small portion of all (sparse) features may be observed in a relatively small fixed set of instances during tuning, we are prone to overfit the training data. An alternative approach for solving this problem is estimating discriminative feature weights directly on the training bi-text (Tillmann and Zhang, 2006; Blunsom et al., 2008; Simianer et al., 2012), which is usually substantially larger than the tuning set, but this is complementary to our goal here of better generalization given a fixed size tuning set.

In order to achieve that goal, we need to carefully choose what objective to optimize, and how to perform parameter estimation of  $\mathbf{w}$  for this objective. We focus on large-margin methods such as SVM (Joachims, 1998) and passive-aggressive algorithms such as MIRA. Intuitively these seek a  $\mathbf{w}$  such that the *separating distance* in geometric space of two hypotheses is at least as large as the *cost* incurred by selecting the incorrect one. This criterion performs well in practice at finding a linear separator in high-dimensional feature spaces (Tsochantaridis et al., 2004; Crammer et al., 2006).

Now, recent advances in machine learning have shown that the generalization ability of these learners can be improved by utilizing second order information, as in the Second Order Perceptron (Cesa-Bianchi et al., 2005), Gaussian Margin Machines (Crammer et al., 2009b), confidence-weighted learning (Dredze and Crammer, 2008), AROW (Crammer et al., 2009a; Chiang, 2012) and Relative Margin Machines (RMM) (Shivaswamy and Jebara, 2009b). The latter, RMM, was introduced as an effective and less computationally expensive way to incorporate the *spread* of the data – second order information about the

distance between hypotheses when projected onto the line defined by the weight vector  $\mathbf{w}$ .

Unfortunately, not all advances in machine learning are easy to apply to structured prediction problems such as SMT; the latter often involve latent variables and surrogate references, resulting in loss functions that have not been well explored in machine learning (Mcalister and Keshet, 2011; Gimpel and Smith, 2012). Although Shivaswamy and Jebara extended RMM to handle sequential structured prediction (Shivaswamy and Jebara, 2009a), their batch approach to quadratic optimization, using existing off-the-shelf QP solvers, does not provide a practical solution: as Taskar et al. (2006) observe, “off-the-shelf QP solvers tend to scale poorly with problem and training sample size” for structured prediction problems. This motivates an online gradient-based optimization approach—an approach that is particularly attractive because its simple update is well suited for efficiently processing structured objects with sparse features (Crammer et al., 2012).

The contributions of this paper include (1) introduction of a loss function for structured RMM in the SMT setting, with surrogate reference translations and latent variables; (2) an online gradient-based solver, RM, with a closed-form parameter update to optimize the relative margin loss; and (3) an efficient implementation that integrates well with the open source cdec SMT system (Dyer et al., 2010).<sup>1</sup> In addition, (4) as our solution is not dependent on any specific QP solver, it can be easily incorporated into practically any gradient-based learning algorithm.

After background discussion on learning in SMT (§2), we introduce a novel online learning algorithm for relative margin maximization suitable for SMT (§3). First, we introduce RMM (§3.1) and propose a latent structured relative margin objective which incorporates cost-augmented hypothesis selection and latent variables. Then, we derive a simple closed-form online update necessary to create a large margin solution while simultaneously bounding the spread of the projection of the data (§3.2). Chinese-English translation experiments show that our algorithm, RM, significantly outperforms strong state-of-the-art optimizers, in both a basic feature setting and high-dimensional (sparse) feature space (§4). Additional Arabic-English experiments further validate these results,

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

even where previously MERT was shown to be advantageous (§5). Finally, we discuss the spread and other key issues of RM (§6), and conclude with discussion of future work (§7).

## 2 Learning in SMT

Given an input sentence in the source language  $x \in \mathcal{X}$ , we want to produce a translation  $y \in \mathcal{Y}(x)$  using a linear model parameterized by a weight vector  $\mathbf{w}$ :

$$(y^*, d^*) = \arg \max_{(y,d) \in \mathcal{Y}(x), \mathcal{D}(x)} \mathbf{w}^\top \mathbf{f}(x, y, d)$$

where  $\mathbf{w}^\top \mathbf{f}(x, y, d)$  is the weighted feature scoring function, hereafter  $s(x, y, d)$ , and  $\mathcal{Y}(x)$  is the space of possible translations of  $x$ . While many derivations  $d \in \mathcal{D}(x)$  can produce a given translation, we are only able to observe  $y$ ; thus we model  $d$  as a latent variable. Although our models are actually defined over derivations, they are always paired with translations, so our feature function  $\mathbf{f}(x, y, d)$  is defined over derivation–translation pairs.<sup>2</sup> The learning goal is then to estimate  $\mathbf{w}$ .

The instability of MERT in larger feature sets (Foster and Kuhn, 2009; Hopkins and May, 2011), has motivated many alternative tuning methods for SMT. These include strategies based on batch log-linear models (Tillmann and Zhang, 2006; Blunsom et al., 2008), as well as the introduction of online linear models (Liang et al., 2006a; Arun and Koehn, 2007).

Recent batch optimizers, PRO and RAMPION, and Batch-MIRA (Cherry and Foster, 2012), have been partly motivated by existing MT infrastructures, as they iterate between decoding the entire tuning set and optimizing the parameters. PRO considers tuning a classification problem and employs a binary classifier to rank pairs of outputs. RAMPION aims to address the disconnect between MT and machine learning by optimizing a structured ramp loss with a concave-convex procedure.

### 2.1 Large-Margin Learning

Online large-margin algorithms, such as MIRA, have also gained prominence in SMT, thanks to their ability to learn models in high-dimensional feature spaces (Watanabe et al., 2007; Chiang et al., 2009). The usual presentation of MIRA’s optimization problem is given as a quadratic program:

<sup>2</sup>We may omit  $d$  in some equations for clarity.

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C\xi_i \quad (1)$$

$$\text{s.t. } s(x_i, y_i, d) - s(x_i, y', d) \geq \Delta_i(y') - \xi_i$$

where  $y'$  is the single most violated constraint, the cost  $\Delta_i(y)$  is computed using an external measure of quality, such as 1-BLEU( $y_i, y$ ), and a slack variable  $\xi_i$  is introduced to allow for non-separable instances.  $C$  acts as a regularization parameter, trading off between margin maximization and constraint violations.

While solving the optimization problem relies on computing the margin between the correct output  $y_i$ , and  $y'$ , in SMT our decoder is often incapable of producing the reference translation, i.e.  $y_i \notin \mathcal{Y}(x_i)$ . We must instead resort to selecting a surrogate reference,  $y^+ \in \mathcal{Y}(x_i)$ . This issue has recently received considerable attention (Liang et al., 2006a; Eidelman, 2012; Chiang, 2012), with preference given to surrogate references obtained through cost-diminished hypothesis selection. Thus,  $y^+$  is selected based on a combination of model score and error metric from the  $k$ -best list produced by our current model. A similar selection is made for the cost-augmented hypothesis  $y^- \in \mathcal{Y}(x_i)$ :

$$(y^+, d^+) \leftarrow \arg \max_{(y,d) \in \mathcal{Y}(x_i), \mathcal{D}(x_i)} s(x_i, y, d) - \Delta_i(y)$$

$$(y^-, d^-) \leftarrow \arg \max_{(y,d) \in \mathcal{Y}(x_i), \mathcal{D}(x_i)} s(x_i, y, d) + \Delta_i(y)$$

In this setting, the optimization problem becomes:

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C\xi_i \quad (2)$$

$$\text{s.t. } \delta s(x_i, y^+, y^-) \geq \Delta_i(y^-) - \Delta_i(y^+) - \xi_i$$

where  $\delta s(x_i, y^+, y^-) = s(x_i, y^+, d^+) - s(x_i, y^-, d^-)$

This leads to a variant of the structured ramp loss to be optimized:

$$\begin{aligned} \ell = & \\ & - \max_{(y^+, d^+) \in \mathcal{Y}(x_i), \mathcal{D}(x_i)} (s(x_i, y^+, d^+) - \Delta_i(y^+)) \\ & + \max_{(y^-, d^-) \in \mathcal{Y}(x_i), \mathcal{D}(x_i)} (s(x_i, y^-, d^-) + \Delta_i(y^-)) \end{aligned} \quad (3)$$

The passive-aggressive update (Crammer et al., 2006), which is used to solve this problem, updates  $\mathbf{w}$  on each round such that the score of the correct hypothesis  $y^+$  is greater than the score of the incorrect  $y^-$  by a margin at least as large as the cost incurred by predicting the incorrect hypothesis, while keeping the change to  $\mathbf{w}$  small.

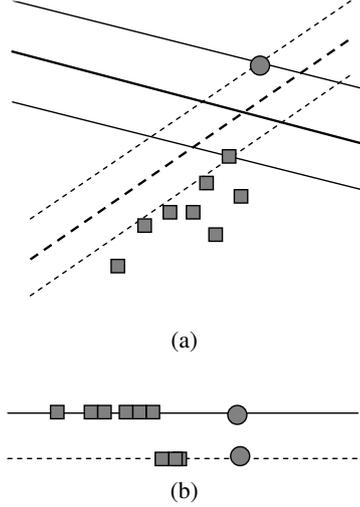


Figure 1: (a) RM and large margin solution comparison and (b) the spread of the projections given by each. RM and large margin solutions are shown with a darker dotted line and a darker solid line, respectively.

### 3 The Relative Margin Machine in SMT

#### 3.1 Relative Margin Machine

The margin, the distance between the correct hypothesis and incorrect one, is defined by  $s(x_i, y^+, d^+)$  and  $s(x_i, y^-, d^-)$ . It is maximized by minimizing the norm in SVM, or analogously, the proximity constraint in MIRA:  $\arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2$ . However, theoretical results supporting large-margin learning, such as the VC-dimension (Vapnik, 1995) or the Rademacher bound (Bartlett and Mendelson, 2003) consider measures of complexity, in addition to the empirical performance, when describing future predictive ability. The measures of complexity usually take the form of some value on the radius of the data, such as the ratio of the radius of the data to the margin (Shivaswamy and Jebara, 2009a). As radius is a way of measuring spread in any projection direction, here we will specifically be interested in the spread of the data as measured after the projection defined by the learned model  $\mathbf{w}$ .

More formally, the *spread* is the distance between  $y^+$ , and the worst candidate  $(y^w, d^w) \leftarrow \arg \min_{(y,d) \in \mathcal{Y}(x_i), \mathcal{D}(x_i)} s(x_i, y, d)$ , after projecting both onto the line defined by the weight vector  $\mathbf{w}$ . For each  $y'$ , this projection is conveniently given by  $s(x_i, y', d)$ , thus the spread is calculated as  $\delta s(x_i, y^+, y^w)$ .

RMM was introduced as a generalization over SVM that incorporates both the margin constraint

and information regarding the spread of the data. The relative margin is the ratio of the absolute, or maximum margin, to the spread of the projected data. Thus, the RMM learns a large margin solution relative to the spread of the data, or in other words, creates a max margin while simultaneously bounding the spread of the projected data. As a concrete example, consider the plot shown in Figure 1(a), with hypotheses represented by two-dimensional feature vectors. The point marked with a circle in the upper right represents  $\mathbf{f}(x_i, y^+)$ , while all other squares represent alternative incorrect hypotheses  $\mathbf{f}(x_i, y')$ . The large margin decision boundary is shown with a darker solid line, while the relative margin solution is shown with a darker dotted line. The lighter lines parallel to each define the margins, with the square at the intersection being  $\mathbf{f}(x_i, y^-)$ . The bottom portion of Figure 1(b) presents an alternative view of each solution, showing the projections of the hypotheses given the learned model of each. Notice that with a large margin solution, although the distance between  $y^+$  and  $y^-$  is greater, the points are highly spread, extending far to the left of the decision boundary.

In contrast, with a relative margin, although we have a smaller absolute margin, the spread is smaller, all points being within a smaller distance  $\epsilon$  of the decision boundary. The higher the spread of the projection, the higher the variance of the projected points, and the greater the likelihood that we will mislabel a new instance, since the high variance projections may cross the learned decision boundary. In higher dimensions, accounting for the spread becomes even more crucial, as will be discussed in Section 6.<sup>3</sup>

Although RMM is theoretically well-founded and improves practical performance over large-margin learning in the settings where it was introduced, it is unsuitable for most complex structured prediction in NLP. Nonetheless, since structured RMM is a generalization of Structured SVM, which shares its underlying objective with MIRA, our intuition is that SMT should be able to benefit as well. But to take advantage of the second-order information RMM utilizes for increased generalizability in SMT, we need a computationally effi-

<sup>3</sup>The motivation of confidence-weighted estimation (Dredze and Crammer, 2008) and AROW (Crammer et al., 2009a) is related in spirit. They use second-order information in the form of a distribution over weights to change the maximum margin solution.

cient optimization procedure that does not require batch training or an off-the-shelf QP solver.

### 3.2 RM Algorithm

We address the above-mentioned limitations by introducing a novel online learning algorithm for relative margin maximization, RM. The relative margin solution is obtained by maximizing the same margin as Equation (2), but now with respect to the distance between  $y^+$ , and the worst candidate  $y^w$ . Thus, the relative margin dictates trading-off between a large margin as before, and a small spread of the projection, in other words, bounding the distance between  $y^+$  and  $y^w$ . The additional computation required, namely, obtaining  $y^w$ , is efficient to perform, and has likely already happened while obtaining the  $k$ -best derivations necessary for the margin update. The online latent structured soft relative margin optimization problem is then:

$$\begin{aligned} \mathbf{w}_{t+1} &= \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C\xi_i + D\tau_i \\ \text{s.t.} \quad & \delta s(x_i, y^+, y^-) \geq \Delta_i(y^-) - \Delta_i(y^+) - \xi_i \\ & -B - \tau_i \leq \delta s(x_i, y^+, y^w) \leq B + \tau_i \end{aligned} \quad (4)$$

where additional bounding constraints are added to the usual margin constraints in order to contain the spread by bounding the difference in projections.  $B$  is an additional parameter; it controls the spread, trading off between margin maximization and spread minimization. Notice that when  $B \rightarrow \infty$ , the bounding constraints disappear, and we are left with the original problem in Equation (2).  $D$ , which plays an analogous role to  $C$ , allows penalized violations of the bounding constraints.

The dual of Equation (4) can be derived as:

$$\begin{aligned} \max_{\alpha, \beta, \beta^*} \mathcal{L} &= \sum_{y \in \mathcal{Y}(x_i)} \alpha_y - B \sum_{y \in \mathcal{Y}(x_i)} \beta_y - B \sum_{y \in \mathcal{Y}(x_i)} \beta_y^* \\ &- \frac{1}{2} \left\langle \sum_{y \in \mathcal{Y}(x_i)} \alpha_y \omega_i(y^+, y) - \sum_{y \in \mathcal{Y}(x_i)} \beta_y \omega_i(y^+, y) \right. \\ &\quad \left. + \sum_{y \in \mathcal{Y}(x_i)} \beta_y^* \omega_i(y^+, y), \right. \\ &\quad \left. \sum_{y' \in \mathcal{Y}(x_j)} \alpha_{y'} \omega_j(y^+, y') - \sum_{y' \in \mathcal{Y}(x_j)} \beta_{y'} \omega_j(y^+, y') \right. \\ &\quad \left. + \sum_{y' \in \mathcal{Y}(x_j)} \beta_{y'}^* \omega_j(y^+, y') \right\rangle \end{aligned} \quad (5)$$

where the  $\alpha$  Lagrange multiplier corresponds to the standard margin constraint, while  $\beta$  and

$\beta^*$  each correspond to a bounding constraint, and  $\omega_i(y^+, y')$  corresponds to the difference of  $f(x_i, y^+, d^+)$  and  $f(x_i, y', d')$ . The weight update can then be obtained from the dual variables:

$$\sum \alpha_y \omega_i(y^+, y) - \sum \beta_y \omega_i(y^+, y) + \sum \beta_y^* \omega_i(y^+, y) \quad (6)$$

The dual in Equation (5) can be optimized using a cutting plane algorithm, an effective method for solving a relaxed optimization problem in the dual, used in Structured SVM, MIRA, and RMM (Tsochantaridis et al., 2004; Chiang, 2012; Shivaswamy and Jebara, 2009a). The cutting plane presented in Alg. 1 decomposes the overall problem into subproblems which are solved independently by creating working sets  $S_i^j$ , which correspond to the largest violations of either the margin constraint, or bounding constraints, and iteratively satisfying the constraints in each set.

The cutting plane in Alg. 1 makes use of the the closed-form gradient-based updates we derived for RM presented in Alg. 2. The updates amount to performing a subgradient descent step to update  $\mathbf{w}$  in accordance with the constraints. Since the constraint matrix of the dual program is not strictly decomposable across constraint types, we are in effect solving an approximation of the original problem.

---

**Algorithm 1** RM Cutting Plane Algorithm (adapted from (Shivaswamy and Jebara, 2009a))

---

**Require:**  $i^{\text{th}}$  training example  $(x_i, y_i)$ , weight  $\mathbf{w}$ , margin reg.  $C$ , bound  $B$ , bound reg.  $D$ ,  $\epsilon$ ,  $\epsilon_B$

- 1:  $S_i^1 \leftarrow \{y^+\}$ ,  $S_i^2 \leftarrow \{y^+\}$ ,  $S_i^3 \leftarrow \{y^+\}$
- 2: **repeat**
- 3:  $H(y) := \Delta_i(y) - \Delta_i(y^+) - \delta s(x_i, y^+, y)$
- 4:  $y_1 \leftarrow \arg \max_{y \in \mathcal{Y}(x_i)} H(y)$
- 5:  $y_2 \leftarrow \arg \max_{y \in \mathcal{Y}(x_i)} G(y) := \delta s(x_i, y^+, y)$
- 6:  $y_3 \leftarrow \arg \min_{y \in \mathcal{Y}(x_i)} -G(y)$
- 7:  $\xi \leftarrow \max\{0, \max_{y \in S_i} H(y)\}$
- 8:  $V_1 \leftarrow H(y_1) - \xi - \epsilon$
- 9:  $V_2 \leftarrow G(y_2) - B - \epsilon_B$
- 10:  $V_3 \leftarrow -G(y_3) - B - \epsilon_B$
- 11:  $j \leftarrow \arg \max_{j' \in \{1,2,3\}} V_{j'}$
- 12: **if**  $V_j > 0$  **then**
- 13:  $S_i^j \leftarrow S_i^j \cup \{y_j\}$
- 14: OPTIMIZE( $\mathbf{w}$ ,  $S_i^1$ ,  $S_i^2$ ,  $S_i^3$ ,  $C$ ,  $B$ )  $\triangleright$  see Alg. 2
- 15: **end if**
- 16: **until**  $S_i^1, S_i^2, S_i^3$  do not change

---

Alternatively, we could utilize a passive-aggressive updating strategy (Crammer et al., 2006), which would simply bypass the cutting plane and select the most violated constraint for

---

**Algorithm 2** RM update with  $\alpha, \beta, \beta^*$

---

- 1: **procedure** OPTIMIZE( $\mathbf{w}$ ,  $S_i^1, S_i^2, S_i^3, C, B$ )
- 2: **while**  $\mathbf{w}$  changes **do**
- 3: **if**  $|S_i^1| > 1$  **then**
- 4: UPDATEMARGIN( $\mathbf{w}$ ,  $S_i^1, C$ )
- 5: **end if**
- 6: **if**  $|S_i^2| > 1$  **then**
- 7: UPDATEUPPERBOUND( $\mathbf{w}$ ,  $S_i^2, B$ )
- 8: **end if**
- 9: **if**  $|S_i^3| > 1$  **then**
- 10: UPDATERLOWERBOUND( $\mathbf{w}$ ,  $S_i^3, B$ )
- 11: **end if**
- 12: **end while**
- 13: **end procedure**
- 14: **procedure** UPDATEMARGIN( $\mathbf{w}$ ,  $S_i^1, C$ )
- 15:  $\alpha_y \leftarrow 0$  **for all**  $y \in S_i^1$
- 16:  $\alpha_{y_i^+} \leftarrow C$
- 17: **for**  $n \leftarrow 1 \dots \text{MaxIter}$  **do**
- 18: Select two constraints  $y, y'$  from  $S_i^1$
- 19:  $\gamma_\alpha \leftarrow \frac{\Delta_i(y') - \Delta_i(y) - \delta s(x_i, y, y')}{\|\omega(y, y')\|^2}$
- 20:  $\gamma_\alpha \leftarrow \max(-\alpha_y, \min(\alpha_{y'}, \gamma_\alpha))$
- 21:  $\alpha_y \leftarrow \alpha_y + \gamma_\alpha$ ;  $\alpha_{y'} \leftarrow \alpha_{y'} - \gamma_\alpha$
- 22:  $\mathbf{w} \leftarrow \mathbf{w} + \gamma_\alpha (\omega(y, y'))$
- 23: **end for**
- 24: **end procedure**
- 25: **procedure** UPDATEUPPERBOUND( $\mathbf{w}$ ,  $S_i^2, B$ )
- 26:  $\beta_y \leftarrow 0$  **for all**  $y \in S_i^2$
- 27: **for**  $n \leftarrow 1 \dots \text{MaxIter}$  **do**
- 28: Select one constraint  $y$  from  $S_i^2$
- 29:  $\gamma_\beta \leftarrow \max(0, \frac{B - \delta s(x_i, y^+, y)}{\|\omega(y^+, y)\|^2})$
- 30:  $\beta_y \leftarrow \beta_y + \gamma_\beta$
- 31:  $\mathbf{w} \leftarrow \mathbf{w} - \gamma_\beta (\omega(y^+, y))$
- 32: **end for**
- 33: **end procedure**
- 34: **procedure** UPDATERLOWERBOUND( $\mathbf{w}$ ,  $S_i^3, B$ )
- 35:  $\beta_y^* \leftarrow 0$  **for all**  $y \in S_i^3$
- 36: **for**  $n \leftarrow 1 \dots \text{MaxIter}$  **do**
- 37: Select one constraint  $y$  from  $S_i^3$
- 38:  $\gamma_{\beta^*} \leftarrow \max(0, \frac{-B - \delta s(x_i, y^+, y)}{\|\omega(y^+, y)\|^2})$
- 39:  $\beta_y^* \leftarrow \beta_y^* + \gamma_{\beta^*}$
- 40:  $\mathbf{w} \leftarrow \mathbf{w} + \gamma_{\beta^*} (\omega(y^+, y))$
- 41: **end for**
- 42: **end procedure**

---

each set, if there is one, and perform the corresponding parameter updates in Alg. 2. We refer to the resulting passive-aggressive algorithm as RM-PA, and the cutting plane version as RM-CP. Preliminary experiments showed that RM-PA performs on par with RM-CP, thus RM-PA is the one used in the empirical evaluation below.

A graphical depiction of the passive-aggressive RM update is presented in Figure 2. The upper right circle represents  $y^+$ , while all other squares represent alternative hypotheses  $y'$ . As in the standard MIRA solution, we select the maximum margin constraint violator,  $y^-$ , shown as the triangle, and update such that the margin is greater than the cost. Additionally, we select the maximum bound-

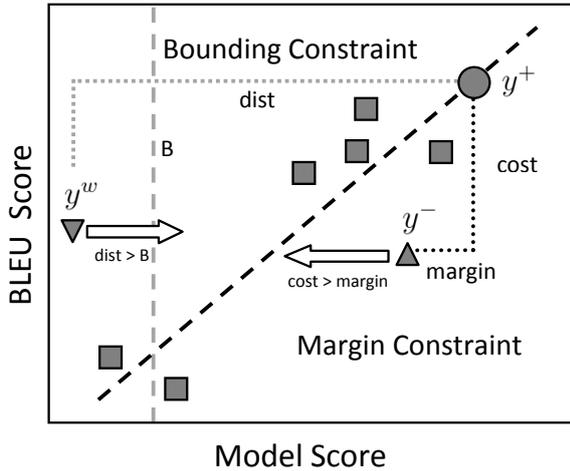


Figure 2: RM update with margin and bounding constraints. The diagonal dotted line depicts cost–margin equilibrium. The vertical gray dotted line depicts the bound  $B$ . White arrows indicate updates triggered by constraint violations. Squares are data points in the  $k$ -best list not selected for update in this round.

task	Corpus	Sentences	Tokens	
			En	Zh/Ar
Zh-En	training	1.6M	44.4M	40.4M
	tune (MT06)	1664	48k	39k
	MT03	919	28k	24k
	MT05	1082	35k	33k
Ar-En	training	1M	23.7M	22.8M
	tune (MT06)	1797	55k	49k
	MT05	1056	36k	33k
	MT08	1360	51k	45k
	4-gram LM	24M	600M	–

Table 1: Corpus statistics

ing constraint violator,  $y^w$ , shown as the upside-down triangle, and update so the distance from  $y^+$  is no greater than  $B$ .

## 4 Experiments

### 4.1 Setup

To evaluate the advantage of explicitly accounting for the spread of the data, we conducted several experiments on two Chinese-English translation test sets, using two different feature sets in each. For training we used the non-UN and non-HK Hansards portions of the NIST training corpora, which was segmented using the Stanford segmenter (Tseng et al., 2005). The data statistics are summarized in the top half of Table 1. The English data was lowercased, tokenized and aligned using GIZA++ (Och and Ney, 2003) to obtain bidirectional alignments, which were symmetrized using the grow-diag-final-and method (Koehn et al., 2003). We trained a 4-gram LM on the

English side of the corpus with additional words from non-NYT and non-LAT, randomly selected portions of the Gigaword v4 corpus, using modified Kneser-Ney smoothing (Chen and Goodman, 1996). We used cdec (Dyer et al., 2010) as our hierarchical phrase-based decoder, and tuned the parameters of the system to optimize BLEU (Papineni et al., 2002) on the NIST MT06 corpus.

We applied several competitive optimizers as baselines: hypergraph-based MERT (Kumar et al., 2009),  $k$ -best variants of MIRA (Crammer et al., 2006; Chiang et al., 2009), PRO (Hopkins and May, 2011), and RAMPION (Gimpel and Smith, 2012). The size of the  $k$ -best list was set to 500 for RAMPION, MIRA and RM, and 1500 for PRO, with both PRO and RAMPION utilizing  $k$ -best aggregation across iterations. RAMPION settings were as described in (Gimpel and Smith, 2012), and PRO settings as described in (Hopkins and May, 2011), with PRO requiring regularization tuning in order to be competitive with the other optimizers. MIRA and RM were run with 15 parallel learners using iterative parameter mixing (McDonald et al., 2010). All optimizers were implemented in cdec and use the same system configuration, thus the only independent variable is the optimizer itself. We set  $C$  to 0.01, and  $MaxIter$  to 100. We selected the bound step size  $D$ , based on performance on a held-out dev set, to be 0.01 for the basic feature set and 0.1 for the sparse feature set. The bound constraint  $B$  was set to 1.<sup>4</sup> The approximate sentence-level BLEU cost  $\Delta_i$  is computed in a manner similar to (Chiang et al., 2009), namely, in the context of previous 1-best translations of the tuning set. All results are averaged over 3 runs.

### 4.2 Feature Sets

We experimented with a small (basic) feature set, and a large (sparse) feature set. For the small feature set, we use 14 features, including a language model, 5 translation model features, penalties for unknown words, the glue rule, and rule arity. For experiments with a larger feature set, we introduced additional lexical and non-lexical sparse Boolean features of the form commonly found in the literature (Chiang et al., 2009; Watan-

<sup>4</sup>We also conducted an investigation into the setting of the  $B$  parameter. We explored alternative values for  $B$ , as well as scaling it by the current candidate’s cost, and found that the optimizer is fairly insensitive to these changes, resulting in only minor differences in BLEU.

Optimizer	Zh	Ar
MIRA	35k	37k
PRO	95k	115k
RAMPION	22k	24k
RM	30k	32k
Active+Inactive	3.4M	4.9M

Table 2: Active sparse feature templates

abe et al., 2007; Simianer et al., 2012).

Non-lexical features include structural distortion, which captures the dependence between re-ordering and the size of a filler, and rule shape, which bins grammar rules by their sequence of terminals and nonterminals (Chiang et al., 2008). Lexical features on rules include rule ID, which fires on a specific grammar rule. We also introduce context-dependent lexical features for the 300 most frequent aligned word pairs  $(f, e)$  in the training corpus, which fire on triples  $(f, e, f_{+1})$  and  $(f, e, f_{-1})$ , capturing when we see  $f$  aligned to  $e$ , with  $f_{+1}$  and  $f_{-1}$  occurring to the right or left of  $f$ , respectively. All other words fall into the default  $\langle unk \rangle$  feature bin. In addition, we have insertion and deletion features for the 150 most frequently unaligned target and source words. These feature templates resulted in a total of 3.4 million possible features, of which only a fraction were active for the respective tuning set and optimizer, as shown in Table 2.

### 4.3 Results

As can be seen from the results in Table 3, our RM method was the best performer in all Chinese-English tests according to all measures – up to 1.9 BLEU and 6.6 TER over MIRA – even though we only optimized for BLEU.<sup>5</sup> Surprisingly, it seems that MIRA did not benefit as much from the sparse features as RM. The results are especially notable for the basic feature setting – up to 1.2 BLEU and 4.6 TER improvement over MERT – since MERT has been shown to be competitive with small numbers of features compared to high-dimensional optimizers such as MIRA (Chiang et al., 2008).

For the tuning set, the decoder performance was consistently the *lowest* with RM, compared to the

<sup>5</sup>In the small feature set RAMPION yielded similar best BLEU scores, but worse TER. In preliminary experiments with a smaller trigram LM, our RM method consistently yielded the highest scores in all Chinese-English tests – up to 1.6 BLEU and 6.4 TER from MIRA, the second best performer.

other optimizers. We believe this is due to the RM bounding constraint being more resistant to overfitting the training data, and thus allowing for improved generalization. Conversely, while PRO had the second lowest tuning scores, it seemed to display signs of underfitting in the basic and large feature settings.

## 5 Additional Experiments

In order to explore the applicability of our approach to a wider range of languages, we also evaluated its performance on Arabic-English translation. All experimental details were the same as above, except those noted below.

For training, we used the non-UN portion of the NIST training corpora, which was segmented using an HMM segmenter (Lee et al., 2003). Dataset statistics are given in the bottom part of Table 1. The sparse feature templates resulted here in a total of 4.9 million possible features, of which again only a fraction were active, as shown in Table 2.

As can be seen in Table 4, in the smaller feature set, RM and MERT were the best performers, with the exception that on MT08, MIRA yielded somewhat better (+0.7) BLEU but a somewhat worse (-0.9) TER score than RM.

On the large feature set, RM is again the best performer, except, perhaps, a tied BLEU score with MIRA on MT08, but with a clear 1.8 TER gain. In both Arabic-English feature sets, MIRA seems to take the second place, while RAMPION lags behind, unlike in Chinese-English (§4).<sup>6</sup>

Interestingly, RM achieved substantially higher BLEU precision scores in all tests for both language pairs. However, this was also usually coupled had a higher brevity penalty (BP) than MIRA, with the BP increasing slightly when moving to the sparse setting.

## 6 Discussion

The trend of the results, summarized as RM gain over other optimizers averaged over all test sets, is presented in Table 5. RM shows clear advantage in both basic and sparse feature sets, over all other state-of-the-art optimizers. The RM gains are notably higher in the large feature set, which we take

<sup>6</sup>In our preliminary experiments with the smaller trigram LM, MERT did better on MT05 in the smaller feature set, and MIRA had a small advantage in two cases. RAMPION performed similarly to RM on the smaller feature set. RM’s loss was only up to 0.8 BLEU (0.7 TER) from MERT or MIRA, while its gains were up to 1.7 BLEU and 2.1 TER over MIRA.

Optimizer	Small (basic) feature set					Large (sparse) feature set				
	Tune	MT03		MT05		Tune	MT03		MT05	
	↑BLEU	↑BLEU	↓TER	↑BLEU	↓TER	↑BLEU	↑BLEU	↓TER	↑BLEU	↓TER
MERT	35.4	35.8	60.8	32.4	63.9	-	-	-	-	-
MIRA	35.5	35.8	61.1	32.1	64.6	36.6	35.9	60.6	32.1	64.1
PRO	34.1	36.0	60.2	31.7	63.4	35.7	34.8	56.1	31.4	59.1
RAMPION	35.1	<b>36.5</b>	58.6	33.0	61.3	36.7	36.9	57.7	33.3	60.6
RM	31.3	<b>36.5</b>	<b>56.4</b>	<b>33.6</b>	<b>59.3</b>	33.2	<b>37.5</b>	<b>54.6</b>	<b>34.0</b>	<b>57.5</b>

Table 3: Performance on Zh-En with basic (left) and sparse (right) feature sets on MT03 and MT05.

Optimizer	Small (basic) feature set					Large (sparse) feature set				
	Tune	MT05		MT08		Tune	MT05		MT08	
	↑BLEU	↑BLEU	↓TER	↑BLEU	↓TER	↑BLEU	↑BLEU	↓TER	↑BLEU	↓TER
MERT	43.8	<b>53.3</b>	<b>40.2</b>	<b>41.0</b>	50.7	-	-	-	-	-
MIRA	43.0	52.8	40.8	<b>41.3</b>	50.6	44.4	53.4	40.1	<b>41.8</b>	50.2
PRO	41.5	51.3	41.5	39.4	51.5	46.8	53.2	40.0	41.4	49.7
RAMPION	42.4	52.0	40.8	40.0	50.8	44.6	52.9	40.4	41.0	50.4
RM	38.5	<b>53.3</b>	<b>39.8</b>	40.6	<b>49.7</b>	43.0	<b>55.3</b>	<b>37.5</b>	<b>41.8</b>	<b>48.4</b>

Table 4: Performance on Ar-En with basic (left) and sparse (right) feature sets on MT05 and MT08.

Optimizer	Small set		Large set	
	BLEU	TER	BLEU	TER
MERT	0.4	2.6	-	-
MIRA	0.5	3.0	1.4	4.3
PRO	1.4	2.9	2.0	1.7
RAMPION	0.6	1.6	1.2	2.8

Table 5: RM gain over other optimizers averaged over all test sets.

as an indication for the importance of bounding the spread.

**Spread analysis:** For RM, the average spread of the projected data in the Chinese-English small feature set was  $0.9 \pm 3.6$  for all tuning iterations, and  $0.7 \pm 2.9$  for the iteration with the highest decoder performance. In comparison, the spread of the data for MIRA was  $5.9 \pm 20.5$  for the best iteration. In the sparse setting, RM had an average spread of  $0.9 \pm 2.4$  for the best iteration, while MIRA had a spread of  $14.0 \pm 31.1$ . Similarly, on Arabic-English, RM had a spread of  $0.7 \pm 2.4$  in the small setting, and  $0.82 \pm 1.4$  in the sparse setting, while MIRA’s spread was  $9.4 \pm 26.8$  and  $11.4 \pm 22.1$ , for the small and sparse settings, respectively. Notice that the average spread for RM stays about the same when moving to higher dimensions, with the variance decreasing in both cases. For MIRA, however, the average spread

increases in both cases, with the variance being much higher than RM. For instance, observe that the spread of MIRA on Chinese grows from 5.9 to 14.0 in the sparse feature setting. While bounding the spread is useful in the low-dimensional setting (0.7-1.5 BLEU gain with RM over MIRA as shown in Table 3), accounting for the spread is even more crucial with sparse features, where MIRA gains only up to 0.1 BLEU, while RM gains 1 BLEU. These results support the claim that our imposed bound  $B$  indeed helps decrease the spread, and that, in turn, lower spread yields better generalization performance.

**Error Analysis:** The inconclusive advantage of RM over MIRA (in BLEU vs. TER scores) on Arabic-English MT08 calls for a closer look. Therefore we conducted a coarse error analysis on 15 randomly selected sentences from MERT, RMM and MIRA, with basic and sparse feature settings for the latter two. This sample yielded 450 data points for analysis: output of the 5 conditions on 15 sentences scored in 6 violation categories. The categories were: function word drop, content word drop, syntactic error (with a reasonable meaning), semantic error (regardless of syntax), word order issues, and function word mis-translation and “hallucination”. The purpose of this analysis was to get a *qualitative* feel for the output of each model, and a better idea as to why we obtained performance improvements. RM no-

ticeably had more word order and excess/wrong function word issues in the basic feature setting than any optimizer. However, RM seemed to benefit the most from the sparse features, as its bad word order rate dropped close to MIRA, and its excess/wrong function word rate dropped below that of MIRA with sparse features (MIRA’s rate actually doubled from its basic feature set). We conjecture both these issues will be ameliorated with syntactic features such as those in Chiang et al. (2008). This correlates with our observation that RM’s overall BLEU score is negatively impacted by the BP, as the BLEU precision scores are noticeably higher.

**K-best:** RM is potentially more sensitive to the size and order of the  $k$ -best list. While MIRA is only concerned with the margin between  $y^+$  and  $y^-$ , RM also accounts for the distance between  $y^+$  and  $y^w$ . It might be the case that a larger  $k$ -best, or revisiting previous strategies for  $y^+$  and  $y^-$  selection, such as bold updating, local updating (Liang et al., 2006b), or max-BLEU updating (Tillmann and Zhang, 2006) might have a greater impact. Also, we only explored several settings of  $B$ , and there remains a continuum of RM solutions that trade off between margin and spread in different ways.

**Active features:** Perhaps contrary to expectation, we did not see evidence of a correlation between the number of active features and optimizer performance. RAMPION, with the fewest features, is the closest performer to RM in Chinese, while MIRA, with a greater number, is the closest on Arabic. We also notice that while PRO had the lowest BLEU scores in Chinese, it was competitive in Arabic with the highest number of features.

## 7 Conclusions and Future Work

We have introduced RM, a novel online margin-based algorithm designed for optimizing high-dimensional feature spaces, which introduces constraints into a large-margin optimizer that bound the spread of the projection of the data while maximizing the margin. The closed-form online update for our relative margin solution accounts for surrogate references and latent variables.

Experimentation in statistical MT yielded significant improvements over several other state-of-the-art optimizers, especially in a high-dimensional feature space (up to 2 BLEU and 4.3 TER on average). Overall, RM achieves the best or

comparable performance according to two scoring methods in two language pairs, with two test sets each, in small and large feature settings. Moreover, across conditions, RM always yielded the best combined TER-BLEU score.<sup>7</sup>

These improvements are achieved using standard, relatively small tuning sets, contrasted with improvements involving sparse features obtained using much larger tuning sets, on the order of hundreds of thousands of sentences (Liang et al., 2006a; Tillmann and Zhang, 2006; Blunsom et al., 2008; Simianer et al., 2012). Since our approach is complementary to scaling up the tuning data, in future work we intend to combine these two methods. In future work we also intend to explore using additional sparse features that are known to be useful in translation, e.g. syntactic features explored by Chiang et al. (2008).

Finally, although motivated by statistical machine translation, RM is a gradient-based method that can easily be applied to other problems. We plan to investigate its utility elsewhere in NLP (e.g. for parsing) as well as in other domains involving high-dimensional structured prediction.

## Acknowledgments

We would like to thank Pannaga Shivaswamy for valuable discussions, and the anonymous reviewers for their comments. Vladimir Eidelman is supported by a National Defense Science and Engineering Graduate Fellowship. This work was also supported in part by the BOLT program of the Defense Advanced Research Projects Agency, Contract HR0011-12-C-0015.

## References

- Abishek Arun and Philipp Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. In *MT Summit XI*.
- Peter L. Bartlett and Shahar Mendelson. 2003. Rademacher and gaussian complexities: risk bounds and structural results. *J. Mach. Learn. Res.*, 3:463–482, March.
- Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, June.

<sup>7</sup>We and other researchers often use  $\frac{1}{2}(\text{TER} - \text{BLEU})$  as a combined SMT quality metric.

- Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. 2005. A second-order perceptron algorithm. *SIAM J. Comput.*, 34(3):640–668, March.
- Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 310–318.
- Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of NAACL*.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Waikiki, Honolulu, Hawaii.
- David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226.
- David Chiang. 2012. Hope and fear for discriminative training of statistical translation models. *J. Machine Learning Research*.
- Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585.
- Koby Crammer, Alex Kulesza, and Mark Dredze. 2009a. Adaptive regularization of weight vectors. In *Advances in Neural Information Processing Systems 22*, pages 414–422.
- Koby Crammer, Mehryar Mohri, and Fernando Pereira. 2009b. Gaussian margin machines. *Journal of Machine Learning Research - Proceedings Track*, 5:105–112.
- Koby Crammer, Mark Dredze, and Fernando Pereira. 2012. Confidence-weighted linear classification for text categorization. *J. Mach. Learn. Res.*, 98888:1891–1926, June.
- Mark Dredze and Koby Crammer. 2008. Confidence-weighted linear classification. In *ICML 08: Proceedings of the 25th international conference on Machine learning*, pages 264–271. ACM.
- Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of ACL System Demonstrations*.
- Vladimir Eidelman. 2012. Optimization strategies for online large-margin learning in machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*.
- George Foster and Roland Kuhn. 2009. Stabilizing minimum error rate training. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 242–249, Athens, Greece, March. Association for Computational Linguistics.
- Kevin Gimpel and Noah A. Smith. 2012. Structured ramp loss minimization for machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics*.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Thorsten Joachims. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In Claire Nédellec and Céline Rouveirol, editors, *European Conference on Machine Learning*, pages 137–142, Berlin. Springer.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, Stroudsburg, PA, USA.
- Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 163–171.
- Young-Suk Lee, Kishore Papineni, Salim Roukos, Osama Emam, and Hany Hassan. 2003. Language model based Arabic word segmentation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 399–406.
- Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006a. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 761–768.
- Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006b. An end-to-end discriminative approach to machine translation. In *Proceedings of the 2006 International Conference on Computational Linguistics (COLING) - the Association for Computational Linguistics (ACL)*.
- David McAllester and Joseph Keshet. 2011. Generalization bounds and consistency for latent structural probit and ramp loss. In J. Shawe-Taylor,

- R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2205–2212.
- Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464, Los Angeles, California.
- Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. In *Computational Linguistics*, volume 29(21), pages 19–51.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.
- Pannagadatta Shivaswamy and Tony Jebara. 2009a. Structured prediction with relative margin. In *In International Conference on Machine Learning and Applications*.
- Pannagadatta K Shivaswamy and Tony Jebara. 2009b. Relative margin machines. In *In Advances in Neural Information Processing Systems 21*. MIT Press.
- Patrick Simianer, Stefan Riezler, and Chris Dyer. 2012. Joint feature selection in distributed stochastic learning for large-scale discriminative training in smt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Jeju Island, Korea, July.
- David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, Sydney, Australia, July. Association for Computational Linguistics.
- Ben Taskar, Simon Lacoste-Julien, and Michael I. Jordan. 2006. Structured prediction, dual extragradient and bregman projections. *J. Mach. Learn. Res.*, 7:1627–1653, December.
- Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical MT. In *Proceedings of the 2006 International Conference on Computational Linguistics (COLING) - the Association for Computational Linguistics (ACL)*.
- Huihsin Tseng, Pi-Chuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter. In *Fourth SIGHAN Workshop on Chinese Language Processing*.
- Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning, ICML '04*.
- Vladimir N. Vapnik. 1995. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.
- Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, June. Association for Computational Linguistics.

# Handling Ambiguities of Bilingual Predicate-Argument Structures for Statistical Machine Translation

Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong  
 National Laboratory of Pattern Recognition, Institute of Automation,  
 Chinese Academy of Sciences, Beijing, China  
 {ffzhai, jjzhang, yzhou, cqzong}@nlpr.ia.ac.cn

## Abstract

Predicate-argument structure (PAS) has been demonstrated to be very effective in improving SMT performance. However, since a source-side PAS might correspond to multiple different target-side PASs, there usually exist many PAS ambiguities during translation. In this paper, we group PAS ambiguities into two types: role ambiguity and gap ambiguity. Then we propose two novel methods to handle the two PAS ambiguities for SMT accordingly: 1) inside context integration; 2) a novel maximum entropy PAS disambiguation (MEPD) model. In this way, we incorporate rich context information of PAS for disambiguation. Then we integrate the two methods into a PAS-based translation framework. Experiments show that our approach helps to achieve significant improvements on translation quality.

## 1 Introduction

Predicate-argument structure (PAS) depicts the relationship between a predicate and its associated arguments, which indicates the skeleton structure of a sentence on semantic level. Basically, PAS agrees much better between two languages than syntax structure (Fung et al., 2006; Wu and Fung, 2009b). Considering that current syntax-based translation models are always impaired by cross-lingual structure divergence (Eisner, 2003; Zhang et al., 2010), PAS is really a better representation of a sentence pair to model the bilingual structure mapping.

However, since a source-side PAS might correspond to multiple different target-side PASs, there usually exist many PAS ambiguities during translation. For example, in Figure 1, (a) and (b) carry the same source-side PAS  $\langle [A0]_1 [Pred(\text{是})]_2 [A1]_3 \rangle$  for Chinese predicate “是”. However, in Figure 1(a), the corresponding target-side-like PAS is  $\langle [X_1] [X_2] [X_3] \rangle$ , while in

Figure 1(b), the counterpart target-side-like PAS<sup>1</sup> is  $\langle [X_2] [X_3] [X_1] \rangle$ . This is because the two PASs play different roles in their corresponding sentences. Actually, Figure 1(a) is an independent PAS, while Figure 1(b) is a modifier of the noun phrase “中国和俄罗斯”. We call this kind of PAS ambiguity *role ambiguity*.

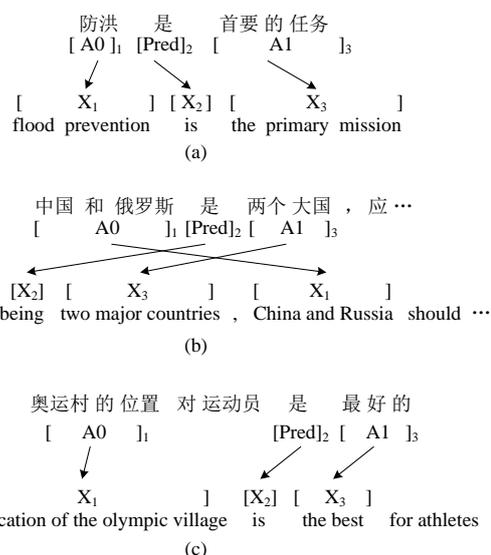


Figure 1. An example of ambiguous PASs.

Meanwhile, Figure 1 also depicts another kind of PAS ambiguity. From Figure 1, we can see that (a) and (c) get the same source-side PAS and target-side-like PAS. However, they are different because in Figure 1(c), there is a gap string “对运动员” between [A0] and [Pred]. Generally, the gap strings are due to the low recall of automatic semantic role labeling (SRL) or complex sentence structures. For example, in Figure 1(c), the gap string “对运动员” is actually an argument “AM-PRP” of the PAS, but the SRL system has

<sup>1</sup>We use *target-side-like PAS* to refer to a list of general non-terminals in target language order, where a non-terminal aligns to a source argument.

ignored it. We call this kind of PAS ambiguity *gap ambiguity*.

During translation, these PAS ambiguities will greatly affect the PAS-based translation models. Therefore, in order to incorporate the bilingual PAS into machine translation effectively, we need to decide which target-side-like PAS should be chosen for a specific source-side PAS. We call this task *PAS disambiguation*.

In this paper, we propose two novel methods to incorporate rich context information to handle PAS ambiguities. Towards the gap ambiguity, we adopt a method called *inside context integration* to extend PAS to IC-PAS. In terms of IC-PAS, the gap strings are combined effectively to deal with the gap ambiguities. As to the role ambiguity, we design a novel *maximum entropy PAS disambiguation (MEPD)* model to combine various context features, such as context words of PAS. For each ambiguous source-side PAS, we build a specific MEPD model to select appropriate target-side-like PAS for translation. We will detail the two methods in Section 3 and 4 respectively.

Finally, we integrate the above two methods into a PAS-based translation framework (Zhai et al. 2012). Experiments show that the two PAS disambiguation methods significantly improve the baseline translation system. The main contribution of this work can be concluded as follows:

- 1) We define two kinds of PAS ambiguities: role ambiguity and gap ambiguity. To our best knowledge, we are the first to handle these PAS ambiguities for SMT.
- 2) Towards the two different ambiguities, we design two specific methods for PAS disambiguation: inside context integration and the novel MEPD model.

## 2 PAS-based Translation Framework

PAS-based translation framework is to perform translation based on PAS transformation (Zhai et al., 2012). In the framework, a source-side PAS is first converted into target-side-like PASs by PAS transformation rules, and then perform translation based on the obtained target-side-like PASs.

### 2.1 PAS Transformation Rules

PAS transformation rules (PASTR) are used to convert a source-side PAS into a target one. Formally, a PASTR is a triple  $\langle Pred, SP, TP \rangle$ :

- *Pred* means the predicate where the rule is extracted.
- *SP* denotes the list of source elements in source language order.
- *TP* refers to the target-side-like PAS, i.e., a list of general non-terminals in target language order.

For example, Figure 2 shows the PASTR extracted from Figure 1(a). In this PASTR, *Pred* is Chinese verb “是”, *SP* is the source element list  $\langle [A0]_1 [Pred]_2 [A1]_3 \rangle$ , and *TP* is the list of non-terminals  $\langle X_1 X_2 X_3 \rangle$ . The same subscript in *SP* and *TP* means a one-to-one mapping between a source element and a target non-terminal. Here, we utilize the source element to refer to the predicate or argument of the source-side PAS.

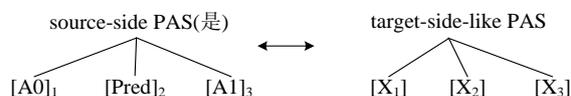


Figure 2. An example PASTR.

### 2.2 PAS Decoding

The PAS decoding process is divided into 3 steps:

(1) **PAS acquisition**: perform semantic role labeling (SRL) on the input sentences to achieve their PASs, i.e., source-side PASs;

(2) **Transformation**: use the PASTR to match the source-side PAS i.e., the predicate *Pred* and the source element list *SP*. Then by the matching PASTRs, transform source-side PASs to target-side-like PASs.

(3) **Translation**: in this step, the decoder first translates each source element respectively, and then a CKY-style decoding algorithm is adopted to combine the translation of each element and get the final translation of the PAS.

### 2.3 Sentence Decoding with the PAS-based translation framework

Sometimes, the source sentence cannot be fully covered by the PAS, especially when there are several predicates. Thus to translate the whole sentence, Zhai et al. (2012) further designed an algorithm to decode the entire sentence.

In the algorithm, they organized the space of translation candidates into a hypergraph. For the span covered by PAS (PAS span), a multiple-branch hyperedge is employed to connect it to the PAS's elements. For the span not covered by PAS (non-PAS span), the decoder considers all the possible binary segmentations of it and utilizes binary hyperedges to link them.

During translation, the decoder fills the spans with translation candidates in a bottom-up manner. For the PAS span, the PAS-based translation framework is adopted. Otherwise, the BTG system (Xiong et al., 2006) is used. When the span covers the whole sentence, we get the final translation result.

Obviously, PAS ambiguities are not considered in this framework at all. The target-side-like PAS is selected only according to the language model and translation probabilities, without considering any context information of PAS. Consequently, it would be difficult for the decoder to distinguish the source-side PAS from different context. This harms the translation quality. Thus to overcome this problem, we design two novel methods to cope with the PAS ambiguities: inside-context integration and a maximum entropy PAS disambiguation (MEPD) model. They will be detailed in the next two sections.

### 3 Inside Context Integration

In this section, we integrate the inside context of the PAS into PASTRs to do PAS disambiguation. Basically, a PAS consists of several elements (a predicate and several arguments), which are actually a series of continuous spans. For a specific PAS  $\langle E_1, \dots, E_n \rangle$ , such as the source-side PAS  $\langle [A0][Pred][A1] \rangle$  in Figure 2, its *controlled range* is defined as:

$$range(PAS) = \{s(E_i), \forall i \in [1, n]\}$$

where  $s(E_i)$  denotes the span of element  $E_i$ . Further, we define the *closure range* of a PAS. It refers to the shortest continuous span covered by the entire PAS:

$$closure\_range = \left[ \min_{j \in s(E_0)} j, \max_{j \in s(E_n)} j \right]$$

Here,  $E_0$  and  $E_n$  are the leftmost and rightmost element of the PAS respectively. The closure range is introduced here because adjacent source elements in a PAS are usually separated by gap strings in the sentence. We call these gap strings the *inside context* (IC) of the PAS, which satisfy:

$$closure\_range(PAS) = \oplus (IC(PAS) \cup range(PAS))$$

The operator  $\oplus$  takes a list of neighboring spans as input<sup>2</sup>, and returns their combined continuous span. As an example, towards the PAS “ $\langle [A0][Pred][A1] \rangle$ ” (the one for Chinese predicate “是(shi)”) in Figure 3, its controlled range is  $\{[3,5],[8,8],[9,11]\}$  and its closure range is  $[3,11]$ . The IC of the PAS is thus  $\{[6,7]\}$ .

To consider the PAS’s IC during PAS transformation process, we incorporate its IC into the extracted PASTR. For each gap string in IC, we abstract it by the sequence of highest node categories (named as s-tag sequence). The s-tag sequence dominates the corresponding syntactic tree fragments in the parse tree. For example, in Figure 3, the s-tag sequence for span  $[6,8]$  is “PP VC”. Thus, the sequence for the IC (span  $[6,7]$ ) in Figure 3 is “PP”. We combine the s-tag sequences with elements of the PAS in order. The resulting PAS is called IC-PAS, just like the left side of Figure 4(b) shows.

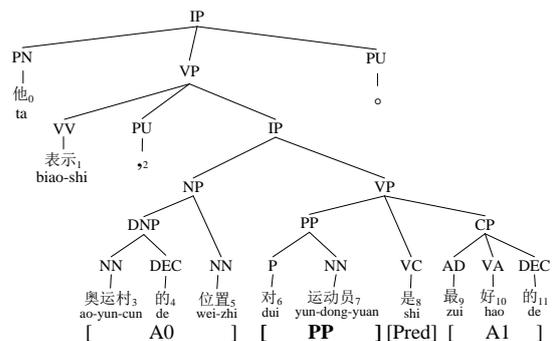


Figure 3. The illustration of inside context (IC). The subscript in each word refers to its position in sentence.

Differently, Zhai et al. (2012) attached the IC to its neighboring elements based on parse trees. For example, in Figure 3, they would attach the gap string “对(dui) 运动员(yun-dong-yuan)” to the PAS’s element “Pred”, and then the span of “Pred” would become  $[6,8]$ . Consequently, the span  $[6,8]$  will be translated as a whole source element in the decoder. This results in a bad translation because the gap string “对(dui) 运动员(yun-dong-yuan)” and predicate “是(shi)” should be translated separately, just as Figure 4(a) shows. Therefore, we can see that the attachment decision in (Zhai et al., 2012) is sometimes unreasonable and the IC also cannot be used for PAS disambiguation at all. In contrast, our meth-

<sup>2</sup> Here, two spans are neighboring means that the beginning of the latter span is the former span’s subsequent word in the sentence. For example, span  $[3,6]$  and  $[7,10]$  are neighboring spans.

od of inside context integration is much flexible and beneficial for PAS disambiguation.

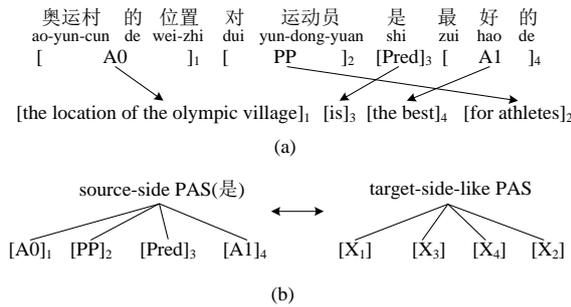


Figure 4. Example of IC-PASTR. (a) The aligned span of each element of the PAS in Figure 3; (b) The extracted IC-PASTR from (a).

Using the IC-PASs, we look for the aligned target span for each element of the IC-PAS. We demand that every element and its corresponding target span must be consistent with word alignment. Otherwise, we discard the IC-PAS. Afterwards, we can easily extract a rule for PAS transformation, which we call IC-PASTR. As an example, Figure 4(b) is the extracted IC-PASTR from Figure 4(a).

Note that we only apply the source-side PAS and word alignment for IC-PASTR extraction. By contrast, Zhai et al. (2012) utilized the result of bilingual SRL (Zhuang and Zong, 2010b). Generally, bilingual SRL could give a better alignment between bilingual elements. However, bilingual SRL usually achieves a really low recall on PASs, about 226,968 entries in our training set while it is 882,702 by using monolingual SRL system. Thus to get a high recall for PASs, we only utilize word alignment instead of capturing the relation between bilingual elements. In addition, to guarantee the accuracy of IC-PASTRs, we only retain rules with more than 5 occurrences.

#### 4 Maximum Entropy PAS Disambiguation (MEPD) Model

In order to handle the role ambiguities, in this section, we concentrate on utilizing a maximum entropy model to incorporate the context information for PAS disambiguation. Actually, the disambiguation problem can be considered as a multi-class classification task. That is to say, for a source-side PAS, every corresponding target-side-like PAS can be considered as a label. For example, in Figure 1, for the source-side PAS “[A0]1[Pred]2[A1]3”, the target-side-like PAS “[X1] [X2] [X3]” in Figure 1(a) is thus a label and

“[X2] [X3] [X1]” in Figure 1(b) is another label of this classification problem.

The maximum entropy model is the classical way to handle this problem:

$$P_{\theta}(tp | sp, c(sp), c(tp)) = \frac{\exp(\sum_i \theta_i h_i(sp, tp, c(sp), c(tp)))}{\sum_{sp'} \exp(\sum_i \theta_i h_i(sp, tp, c(sp), c(tp)))}$$

where  $sp$  and  $tp$  refer to the source-side PAS (not including the predicate) and the target-side-like PAS respectively.  $c(sp)$  and  $c(tp)$  denote the surrounding context of  $sp$  and  $tp$ .  $h_i$  is a binary feature function and  $\theta_i$  is the weight of  $h_i$ .

We train a maximum entropy classifier for each  $sp$  via the off-the-shelf MaxEnt toolkit<sup>3</sup>. Note that to avoid sparseness,  $sp$  does not include predicate of the PAS. Practically, the predicate serves as a feature of the MEPD model. As an example, for the rule illustrated in Figure 4(b), we build a MEPD model for its source element list  $sp <[A0] [PP] [Pred] [A1]>$ , and integrate the predicate “是(shi)” into the MEPD model as a feature.

In detail, we design a list of features for each pair  $<sp, tp>$  as follows:

- **Lexical Features.** These features include the words immediately to the left and right of  $sp$ , represented as  $w_{-1}$  and  $w_{+1}$ . Moreover, the head word of each argument also serves as a lexical feature, named as  $hw(E_i)$ . For example, Figure 3 shows the context of the IC-PASTR in Figure 4(b), and the extracted lexical features of the instance are:  $w_{-1} =$ ,  $w_{+1} =$ ,  $hw([A0]_1) =$ 位置(wei-zhi),  $hw([A1]_4) =$ 好(hao).

- **POS Features.** These features are defined as the POS tags of the lexical features,  $p_{-1}$ ,  $p_{+1}$  and  $phw(E_i)$  respectively. Thus, the corresponding POS features of Figure 4 (b) are:  $p_{-1} =$ PU,  $p_{+1} =$ PU,  $phw([A0]_1) =$ NN,  $phw([A1]_4) =$ VA.

- **Predicate Feature.** It is the pair of source predicate and its corresponding target predicate. For example, in Figure 4(b), the source and target predicate are “是(shi)” and “is” respectively. The predicate feature is thus “PredF=是(shi)+is”. The target predicate is determined by:

$$t-pred = \arg \max_{j \in \text{range}(PAS)} p(t_j | s-pred)$$

where  $s-pred$  is the source predicate and  $t-pred$  is the corresponding target predicate.

<sup>3</sup>[http://homepages.inf.ed.ac.uk/lzhang10/maxent\\_toolkit.htm](http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.htm)

$t\_range(PAS)$  refers to the target range covering all the words that are reachable from the PAS via word alignment.  $t_j$  refers to the  $j$ th word in  $t\_range(PAS)$ . The utilized lexical translation probabilities are from the toolkit in Moses (Koehn et al., 2007).

● **Syntax Features.** These features include  $st(Ei)$ , i.e., the highest syntax tag for each argument, and  $fst(PAS)$  which is the lowest father node of  $sp$  in the parse tree. For example, for the rule shown in Figure 4(b), syntax features are  $st([A0]_1)=NP$ ,  $st([A1]_4)=CP$ , and  $fst(PAS)=IP$  respectively.

Using these features, we can train the MEPD model. We set the Gaussian prior to 1.0 and perform 100 iterations of the L-BFGS algorithm for each MEPD model. At last, we build 160 and 215 different MEPD classifiers, respectively, for the PASTRs and IC-PASTRs. Note that since the training procedure of maximum entropy classifier is really fast, it does not take much time to train these classifiers.

## 5 Integrating into the PAS-based Translation Framework

In this section, we integrate our method of PAS disambiguation into the PAS-based translation framework when translating each test sentence.

For inside context integration, since the format of IC-PASTR is the same to PASTR<sup>4</sup>, we can use the IC-PASTR to substitute PASTR for building a PAS-based translation system directly. We use “IC-PASTR” to denote this system. In addition, since our method of rule extraction is different from (Zhai et al., 2012), we also use PASTR to construct a translation system as the baseline system, which we call “PASTR”.

On the basis of PASTR and IC-PASTR, we further integrate our MEPD model into translation. Specifically, we take the score of the MEPD model as another informative feature for the decoder to distinguish good target-side-like PASs from bad ones. The weights of the MEPD feature can be tuned by MERT (Och, 2003) together with other translation features, such as language model.

## 6 Related Work

The method of PAS disambiguation for SMT is relevant to the previous work on context depend-

ent translation.

Carpuat and Wu (2007a, 2007b) and Chan et al. (2007) have integrated word sense disambiguation (WSD) and phrase sense disambiguation (PSD) into SMT systems. They combine rich context information to do disambiguation for words or phrases, and achieve improved translation performance.

Differently, He et al. (2008), Liu et al. (2008) and Cui et al. (2010) designed maximum entropy (ME) classifiers to do better rule section for hierarchical phrase-based model and tree-to-string model respectively. By incorporating the rich context information as features, they chose better rules for translation and yielded stable improvements on translation quality.

Our work differs from the above work in the following two aspects: 1) in our work, we focus on the problem of disambiguates on PAS; 2) we define two kinds of PAS ambiguities: role ambiguity and gap ambiguity. 3) towards the two different ambiguities, we design two specific methods for PAS disambiguation: inside context integration and the novel MEPD model.

In addition, Xiong et al. (2012) proposed an argument reordering model to predict the relative position between predicates and arguments. They also combine the context information in the model. But they only focus on the relation between the predicate and a specific argument, rather than the entire PAS. Different from their work, we incorporate the context information to do PAS disambiguation based on the entire PAS. This is very beneficial for global reordering during translation (Zhai et al., 2012).

## 7 Experiment

### 7.1 Experimental Setup

We perform Chinese-to-English translation to demonstrate the effectiveness of our PAS disambiguation method. The training data contains about 260K sentence pairs<sup>5</sup>. To get accurate SRL results, we ensure that the length of each sentence in the training data is among 10 and 30 words. We run GIZA++ and then employ the *grow-diag-final-and* (*gdfa*) strategy to produce symmetric word alignments. The development set and test set come from the NIST evaluation test data (from 2003 to 2005). Similar to the training set, we also only retain the sentences

<sup>4</sup> The only difference between IC-PASTR and PASTR is that there are many syntactic labels in IC-PASTRs.

<sup>5</sup> It is extracted from the LDC corpus. The LDC category number: LDC2000T50, LDC2002E18, LDC2003E07, LDC2004T07, LDC2005T06, LDC2002L27, LDC2005T10 and LDC2005T34.

whose lengths are among 10 and 30 words. Finally, the development set includes 595 sentences from NIST MT03 and the test set contains 1,786 sentences from NIST MT04 and MT05.

We train a 5-gram language model with the Xinhua portion of English Gigaword corpus and target part of the training data. The translation quality is evaluated by case-insensitive BLEU-4 with shortest length penalty. The statistical significance test is performed by the re-sampling approach (Koehn, 2004).

We perform SRL on the source part of the training set, development set and test set by the Chinese SRL system used in (Zhuang and Zong, 2010b). To relieve the negative effect of SRL errors, we get the multiple SRL results by providing the SRL system with 3-best parse trees of Berkeley parser (Petrov and Klein, 2007), 1-best parse tree of Bikel parser (Bikel, 2004) and Stanford parser (Klein and Manning, 2003). Therefore, at last, we can get 5 SRL result for each sentence. For the training set, we use these SRL results to do rule extraction respectively. We combine the obtained rules together to get a combined rule set. We discard the rules with fewer than 5 appearances. Using this set, we can train our MEPD model directly.

As to translation, we match the 5 SRL results with transformation rules respectively, and then apply the resulting target-side-like PASs for decoding. As we mentioned in section 2.3, we use the state-of-the-art BTG system to translate the non-PAS spans.

source-side PAS	counts	number of classes
[A0] [Pred(是)] [A1]	245	6
[A0] [Pred(说)] [A1]	148	6
[A0] [AM-ADV] [Pred(是)] [A1]	68	20
[A0] [Pred(表示)] [A1]	66	6
[A0] [Pred(有)] [A1]	42	6
[A0] [Pred(认为)] [A1]	32	4
[A0] [AM-ADV] [Pred(有)] [A1]	32	19
[A0] [Pred(指出)] [A1]	29	4
[AM-ADV] [Pred(有)] [A1]	26	6
[A2] [Pred(为)] [A1]	16	5

Table 1. The top 10 frequent source-side PASs in the dev and test set.

## 7.2 Ambiguities in Source-side PASs

We first give Table 1 to show some examples of role ambiguity. In the table, for instance, the second line denotes that the source-side PAS “[A0] [Pred(说)] [A1]” appears 148 times in the devel-

opment and test set all together, and it corresponds to 6 different target-side-like PASs in the training set.

As we can see from Table 1, all the top 10 PASs correspond to several different target-side-like PASs. Moreover, according to our statistics, among all PASs appearing in the development set and test set, 56.7% of them carry gap strings. These statistics demonstrate the importance of handling the role ambiguity and gap ambiguity in the PAS-based translation framework. Therefore, we believe that our PAS disambiguation method would be helpful for translation.

## 7.3 Translation Result

We compare the translation result using PASTR, IC-PASTR and our MEPD model in this section. The final translation results are shown in Table 2. As we can see, after employing PAS for translation, all systems outperform the baseline BTG system significantly. This comparison verifies the conclusion of (Zhai et al., 2012) and thus also demonstrates the effectiveness of PAS.

MT system	Test set	n-gram precision			
		1	2	3	4
BTG	32.75	74.39	41.91	24.75	14.91
PASTR	33.24*	75.28	42.62	25.18	15.10
PASTR+MEPD	33.78*	75.32	43.08	25.75	15.58
IC-PASTR	33.95*#	75.62	43.36	25.92	15.58
IC-PASTR+MEPD	<b>34.19*#</b>	75.66	43.40	26.15	15.92

Table 2. Result of baseline system and the MT systems using our PAS-based disambiguation method. The “\*” and “#” denote that the result is significantly better than BTG and PASTR respectively ( $p < 0.01$ ).

Specifically, after integrating the inside context information of PAS into transformation, we can see that system IC-PASTR significantly outperforms system PASTR by 0.71 BLEU points. Moreover, after we import the MEPD model into system PASTR, we get a significant improvement over PASTR (by 0.54 BLEU points). These comparisons indicate that both the inside context integration and our MEPD model are beneficial for the decoder to choose better target-side-like PAS for translation.

On the basis of IC-PASTR, we further add our MEPD model into translation and get system IC-PASTR+MEPD. We can see that this system further achieves a remarkable improvement over system PASTR (0.95 BLEU points).

However, from Table 2, we find that system IC-PASTR+MEPD only outperforms system IC-PASTR slightly (0.24 BLEU points). The result seems to show that our MEPD model is not such

useful after using IC-PASTR. We will explore the reason in section 7.5.

### 7.4 Effectiveness of Inside Context Integration

The method of inside context integration is used to combine the inside context (gap strings) into PAS for translation, i.e., extend the PASTR to IC-PASTR. In order to demonstrate the effectiveness of inside context integration, we first give Table 3, which illustrates statistics on the matching PASs. The statistics are conducted on the combination of development set and test set.

Transformation Rules	Matching PAS		
	None Gap PAS	Gap PAS	Total
PASTR	1702	1539	3241
IC-PASTR	1546	832	2378

Table 3. Statistics on the matching PAS.

In Table 3, for example, the line for PASTR means that if we use PASTR for the combined set, 3241 PASs (column “Total”) can match PASTRs in total. Among these matching PASs, 1539 ones (column “Gap PAS”) carry gap strings, while 1702 ones do not (column “None Gap PAS”). Consequently, since PASTR does not consider the inside context during translation, the Gap PASs, which account for 47% (1539/3241) of all matching PASs, might be handled inappropriately in the PAS-based translation framework. Therefore, integrating the inside context into PASTRs, i.e., using the proposed IC-PASTRs, would be helpful for translation. The translation result shown in Table 2 also demonstrates this conclusion.

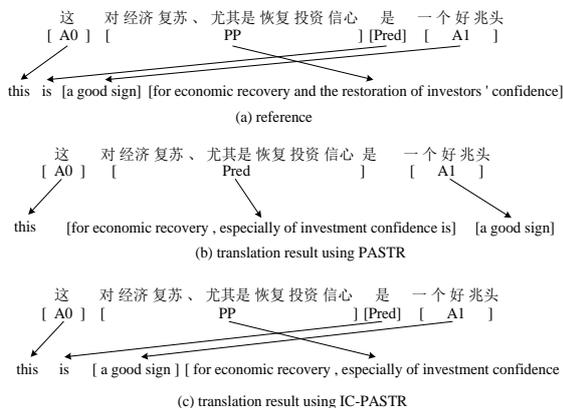
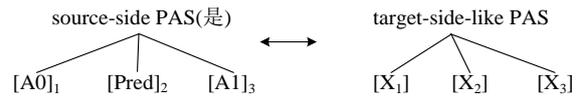


Figure 5. Translation examples to verify the effectiveness of inside context.

From Table 3, we can also find that the number of matching PASs decreases after using IC-PASTR. This is because IC-PASTR is more spe-

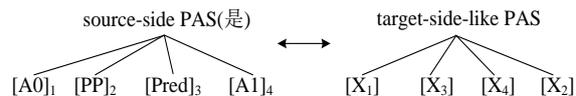
cific than PASTR. Therefore, for a PAS with specific inside context (gap strings), even if the matched PASTR is available, the matched IC-PASTR might not. This indicates that comparing with PASTR, IC-PASTR is more capable of distinguishing different PASs. Therefore, based on this advantage, although the number of matching PASs decreases, IC-PASTR still improves the translation system using PASTR significantly. Of course, we believe that it is also possible to integrate the inside context without decreasing the number of matching PASs and we plan this as our future work.

We further give a translation example in Figure 5 to illustrate the effectiveness of our inside context integration method. In the example, the automatic SRL system ignores the long preposition phrase “对经济复苏、尤其是恢复投资信心” for the PAS. Thus, the system using PASTRs can only attach the long phrase to the predicate “是” according to the parse tree, and meanwhile, make use of a transformation rule as follows:



In this way, the translation result is very bad, just as Figure 5(b) shows. The long preposition phrases are wrongly positioned in the translation.

In contrast, after inside context integration, we match the inside context during PAS transformation. As Figure 5(c) shows, the inside context helps to select a right transformation rule as follows and gets a good translation result finally.



### 7.5 Effectiveness of the MEPD Model

The MEPD model incorporates various context features to select better target-side-like PAS for translation. On the basis of PASTR and IC-PASTR, we build 160 and 215 different MEPD classifiers, respectively, for the frequent source-side PASs.

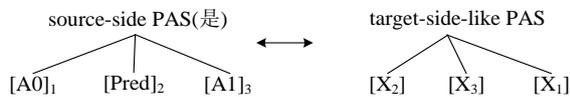
In Table 2, we have found that our MEPD model improves system IC-PASTR slightly. We conjecture that this phenomenon is due to two possible reasons. On one hand, sometimes, many PAS ambiguities might be resolved by both inside context and the MEPD model. Therefore, the improvement would not be such significant

when we combine these two methods together. On the other hand, as Table 3 shows, the number of matching PASs decreases after using IC-PASTR. Since the MEPD model works on PASs, its effectiveness would also weaken to some extent. Future work will explore this phenomenon more thoroughly.

Ref	<p>... , [海牙] [是] [其 最后 一站] 。</p> <p>... [the hague] [is] [his last stop] .</p>
PASTR	<p>... , [海牙]<sub>A0</sub> [是]<sub>Pred</sub> [其 最后 一站]<sub>A1</sub> 。</p> <p>... [is] [his last leg of] [the hague] .</p>
PASTR + MEPD	<p>... , [海牙]<sub>A0</sub> [是]<sub>Pred</sub> [其 最后 一站]<sub>A1</sub> 。</p> <p>... [the hague] [is] [the last leg] .</p>

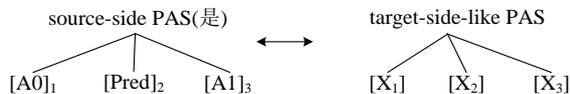
Figure 6. Translation examples to demonstrate the effectiveness of our MEPD model.

Now, we give Figure 6 to demonstrate the effectiveness of our MEPD model. From the Figure, we can see that the system using PASTRs selects an inappropriate transformation rule for translation:



This rule wrongly moves the subject “海牙 (Hague)” to the end of the translation. We do not give the translation result of the BTG system here because it makes the same mistake.

Conversely, by considering the context information, the PASTR+MEPD system chooses a correct rule for translation:



As we can see, the used rule helps to keep the SVO structure unchanged, and gets the correct translation.

## 8 Conclusion and Future Work

In this paper, we focus on the problem of ambiguities for PASs. We first propose two ambiguities: gap ambiguity and role ambiguity. Accordingly, we design two novel methods to do efficient PAS disambiguation: inside-context integration and a novel MEPD model. For inside context integration, we abstract the inside con-

text and combine them into the PASTRs for PAS transformation. Towards the MEPD model, we design a maximum entropy model for each ambitious source-side PASs. The two methods successfully incorporate the rich context information into the translation process. Experiments show that our PAS disambiguation methods help to improve the translation performance significantly.

In the next step, we will conduct experiments on other language pairs to demonstrate the effectiveness of our PAS disambiguation method. In addition, we also will try to explore more useful and representative features for our MEPD model.

## Acknowledgments

The research work has been funded by the Hi-Tech Research and Development Program (“863” Program) of China under Grant No. 2011AA01A207, 2012AA011101, and 2012AA011102 and also supported by the Key Project of Knowledge Innovation Program of Chinese Academy of Sciences under Grant No.KGZD-EW-501. We thank the anonymous reviewers for their valuable comments and suggestions.

## References

- Wilker Aziz, Miguel Rios, and Lucia Specia. (2011). Shallow semantic trees for smt. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 316–322, Edinburgh, Scotland, July.
- Daniel Bikel. (2004). Intricacies of Collins parsing model. *Computational Linguistics*, 30(4):480-511.
- David Chiang. (2007). Hierarchical phrase-based translation. *Computational Linguistics*, 33 (2):201–228.
- Marine Carpuat and Dekai Wu. 2007a. How phrase-sense disambiguation outperforms word sense disambiguation for statistical machine translation. In *11th Conference on Theoretical and Methodological Issues in Machine Translation*, pages 43–52.
- Marine Carpuat and Dekai Wu. 2007b. Improving statistical machine translation using word sense disambiguation. In *Proceedings of EMNLP-CoNLL 2007*, pages 61–72.
- Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proc. ACL 2007*, pages 33–40.
- Lei Cui, Dongdong Zhang, Mu Li, Ming Zhou and Tiejun Zhao. A Joint Rule Selection Model for Hierarchical Phrase-Based Translation. In *Proc. of ACL 2010*.

- Jason Eisner. (2003). Learning non-isomorphic tree mappings for machine translation. In *Proc. of ACL 2003*.
- Pascale Fung, Wu Zhaojun, Yang Yongsheng, and Dekai Wu. (2006). Automatic learning of chinese english semantic structure mapping. In *IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*, Aruba, December.
- Pascale Fung, Zhaojun Wu, Yongsheng Yang and Dekai Wu. (2007). Learning bilingual semantic frames: shallow semantic parsing vs. semantic sole projection. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation*, pages 75-84.
- Qin Gao and Stephan Vogel. (2011). Utilizing target-side semantic role labels to assist hierarchical phrase-based machine translation. In *Proceedings of Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 107–115, Portland, Oregon, USA, June 2011. Association for Computational Linguistics
- Zhongjun He, Qun Liu, and Shouxun Lin. 2008. Improving statistical machine translation using lexicalized rule selection. In *Proc. of Coling 2008*, pages 321–328.
- Franz Josef Och. (2003). Minimum error rate training in statistical machine translation. In *Proc. of ACL 2003*, pages 160-167.
- Franz Josef Och and Hermann Ney. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417-449.
- Dan Klein and Christopher D. Manning. (2003). Accurate unlexicalized parsing. In *Proc. of ACL-2003*, pages 423-430.
- Philipp Koehn, Franz Joseph Och, and Daniel Marcu. (2003). Statistical phrase-based translation. In *Proceedings of NAACL 2003*, pages 58–54, Edmonton, Canada, May-June.
- Philipp Koehn. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July.
- P Koehn, H Hoang, A Birch, C Callison-Burch, M Federico, N Bertoldi, B Cowan, W Shen, C Moran and R Zens, (2007). Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL 2007*. pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.
- Mamoru Komachi and Yuji Matsumoto. (2006). Phrase reordering for statistical machine translation based on predicate-argument structure. In *Proceedings of the International Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation*, pages 77–82.
- Ding Liu and Daniel Gildea. (2008). Improved tree-to-string transducer for machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 62–69, Columbus, Ohio, USA, June 2008.
- Ding Liu and Daniel Gildea. (2010). Semantic role features for machine translation. In *Proc. of Coling 2010*, pages 716–724, Beijing, China, August.
- Qun Liu, Zhongjun He, Yang Liu, and Shouxun Lin. Maximum Entropy based Rule Selection Model for Syntax-based Statistical Machine Translation. In *Proc. of EMNLP 2008*.
- Yang Liu, Qun Liu and Shouxun Lin. (2006). Tree-to-string alignment template for statistical machine translation. In *Proc. of ACL-COLING 2006*.
- Daniel Marcu, Wei Wang, Abdessamad Echihabi and Kevin Knight. (2006). SPMT: Statistical machine translation with syntactified target language phrases. In *Proc. of EMNLP 2006*, pages 44-52.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proc. ACL 2002*, pages 311–318, Philadelphia, Pennsylvania, USA, July.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July. Association for Computational Linguistics.
- Andreas Stolcke. (2002). Srilm – an extensible language modelling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 901–904, Denver, Colorado, USA, September.
- Dekai Wu and Pascale Fung. (2009a). Can semantic role labelling improve smt. In *Proceedings of the 13th Annual Conference of the EAMT*, pages 218–225, Barcelona, May.
- Dekai Wu and Pascale Fung. (2009b). Semantic roles for smt: A hybrid two-pass model. In *Proc. NAACL 2009*, pages 13–16, Boulder, Colorado, June.
- ShuminWu and Martha Palmer. (2011). Semantic mapping using automatic word alignment and semantic role labelling. In *Proceedings of Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 21–30, Portland, Oregon, USA, June 2011.
- Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. (2011). Extracting preordering rules from predicate-argument structures. In *Proc. IJCNLP 2011*, pages 29–37, Chiang Mai, Thailand, November.

- Deyi Xiong, Qun Liu, and Shouxun Lin. (2006). Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, Australia, July.
- Deyi Xiong, Min Zhang, and Haizhou Li. (2012). Modelling the translation of predicate-argument structure for smt. In *Proc. of ACL 2012*, pages 902–911, Jeju, Republic of Korea, 8-14 July 2012.
- Nianwen Xue. (2008). Labelling chinese predicates with semantic roles. *Computational Linguistics*, 34(2): 225-255.
- Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong. Machine Translation by Modeling Predicate-Argument Structure Transformation. In *Proc. of COLING 2012*.
- Hui Zhang, Min Zhang, Haizhou Li and Eng Siong Chng. (2010). Non-isomorphic Forest Pair Translation. In *Proceedings of EMNLP 2010*, pages 440-450, Massachusetts, USA, 9-11 October 2010.
- Tao Zhuang, and Chengqing Zong. (2010a). A minimum error weighting combination strategy for chinese semantic role labelling. In *Proceedings of COLING-2010*, pages 1362-1370.
- Tao Zhuang and Chengqing Zong. (2010b). Joint inference for bilingual semantic role labelling. In *Proceedings of EMNLP 2010*, pages 304–314, Massachusetts, USA, 9-11 October 2010.

# Reconstructing an Indo-European Family Tree from Non-native English texts

Ryo Nagata<sup>1,2</sup> Edward Whittaker<sup>3</sup>

<sup>1</sup>Konan University / Kobe, Japan

<sup>2</sup>LIMSI-CNRS / Orsay, France

<sup>3</sup>Inferret Limited / Northampton, England

nagata-acl@hyogo-u.ac.jp, ed@inferret.co.uk

## Abstract

Mother tongue interference is the phenomenon where linguistic systems of a mother tongue are transferred to another language. Although there has been plenty of work on mother tongue interference, very little is known about how strongly it is transferred to another language and about what relation there is across mother tongues. To address these questions, this paper explores and visualizes mother tongue interference preserved in English texts written by Indo-European language speakers. This paper further explores linguistic features that explain why certain relations are preserved in English writing, and which contribute to related tasks such as native language identification.

## 1 Introduction

Transfer of linguistic systems of a mother tongue to another language, namely *mother tongue interference*, is often observable in the writing of non-native speakers. The reader may be able to determine the mother tongue of the writer of the following sentence from the underlined article error:

*The alien wouldn't use my spaceship but the hers.*

The answer would probably be French or Spanish; the definite article is allowed to modify possessive pronouns in these languages, and the usage is sometimes negatively transferred to English writing. Researchers such as Swan and Smith (2001), Aarts and Granger (1998), Davidsen-Nielsen and Harder (2001), and Altenberg and Tapper (1998) work on mother tongue interference to reveal overused/underused words, part of speech (POS), or grammatical items.

In contrast, very little is known about how strongly mother tongue interference is transferred to another language and about what relation there is across mother tongues. At one extreme, one could argue that it is so strongly transferred to texts in another language that the linguistic relations between mother tongues are perfectly preserved in the texts. At the other extreme, one can counter it, arguing that other features such as non-nativeness are more influential than mother tongue interference. One possible reason for this is that a large part of the distinctive language systems of a mother tongue may be eliminated when transferred to another language from a speaker's mother tongue. For example, Slavic languages have a rich inflectional case system (e.g., Czech has seven inflectional cases) whereas French does not. However, the difference in the richness cannot be transferred into English because English has almost no inflectional case system. Thus, one cannot determine the mother tongue of a given non-native text from the inflectional case. A similar argument can be made about some parts of gender, tense, and aspect systems. Besides, Wong and Dras (2009) show that there are no significant differences, between mother tongues, in the misuse of certain syntactic features such as subject-verb agreement that have different tendencies depending on their mother tongues. Considering these, one could not be so sure which argument is correct. In any case, to the best of our knowledge, no one has yet answered this question.

In view of this background, we take the first step in addressing this question. We hypothesize that:

**Hypothesis:** Mother tongue interference is so strong that the relations in a language family are preserved in texts written in another language.

In other words, mother tongue interference is so strong that one can reconstruct a language fam-

ily tree from non-native texts. One of the major contributions of this work is to reveal and visualize a language family tree preserved in non-native texts, by examining the hypothesis. This becomes important in native language identification<sup>1</sup> which is useful for improving grammatical error correction systems (Chodorow et al., 2010) or for providing more targeted feedback to language learners. As we will see in Sect. 6, this paper reveals several crucial findings that contribute to improving native language identification. In addition, this paper shows that the findings could contribute to reconstruction of language family trees (Enright and Kondrak, 2011; Gray and Atkinson, 2003; Barbañon et al., 2007; Batagelj et al., 1992; Nakhleh et al., 2005), which is one of the central tasks in historical linguistics.

The rest of this paper is structured as follows. Sect. 2 introduces the basic approach of this work. Sect. 3 discusses the methods in detail. Sect. 4 describes experiments conducted to investigate the hypothesis. Sect. 5 discusses the experimental results. Sect. 6 discusses implications for work in related domains.

## 2 Approach

To examine the hypothesis, we reconstruct a language family tree from English texts written by non-native speakers of English whose mother tongue is one of the Indo-European languages (Beekes, 2011; Ramat and Ramat, 2006). If the reconstructed tree is sufficiently similar to the original Indo-European family tree, it will support the hypothesis. If not, it suggests that some features other than mother tongue interference are more influential.

The approach we use for reconstructing a language family tree is to apply agglomerative hierarchical clustering (Han and Kamber, 2006) to English texts written by non-native speakers. Researchers have already performed related work on reconstructing language family trees. For instance, Kroeber and Chriétien (1937) and Ellegård (1959) proposed statistical methods for measuring the similarity metric between languages. More recently, Batagelj et al. (1992) and Kita (1999) proposed methods for reconstructing language family trees using clustering. Among them, the

<sup>1</sup>Recently, native language identification has drawn the attention of NLP researchers. For instance, a shared task on native language identification took place at an NAACL-HLT 2013 workshop.

most related method is that of Kita (1999). In his method, a variety of languages are modeled by their spelling systems (i.e., character-based  $n$ -gram language models). Then, agglomerative hierarchical clustering is applied to the language models to reconstruct a language family tree. The similarity used for clustering is based on a divergence-like distance between two language models that was originally proposed by Juang and Rabiner (1985). This method is purely data-driven and does not require human expert knowledge for the selection of linguistic features.

Our work closely follows Kita’s work. However, it should be emphasized that there is a significant difference between the two. Kita’s work (and other previous work) targets clustering of a variety of languages whereas our work tries to reconstruct a language family tree preserved in non-native English. This significant difference prevents us from directly applying techniques in the literature to our task. For instance, Batagelj et al. (1992) use basic vocabularies such as *belly* in English and *ventre* in French to measure similarity between languages. Obviously, this does not work on our task; *belly* is *belly* in English writing whoever writes it. Kita’s method is also likely not to work well because all texts in our task share the same spelling system (i.e., English spelling). Although spelling is sometimes influenced by mother tongues, it involves a lot more including overuse, underuse, and misuse of lexical, grammatical, and syntactic systems.

To solve the problem, this work adopts a word-based language model in the expectation that word sequences reflect mother tongue interference. At the same time, its simple application would cause a serious side effect. It would reflect the topics of given texts rather than mother tongue interference. Unfortunately, there exists no such English corpus that covers a variety of language speakers with uniform topics; moreover the availability of non-native corpora is still somewhat limited. This also means that available non-native corpora may be too small to train reliable word-based language models. The next section describes two methods (language model-based and vector-based), which address these problems.

## 3 Methods

### 3.1 Language Model-based Method

To begin with, let us define the following symbols used in the methods. Let  $D_i$  be a set of English

texts where  $i$  denotes a mother tongue  $i$ . Similarly, let  $M_i$  be a language model trained using  $D_i$ .

To solve the problems pointed out in Sect. 2, we use an  $n$ -gram language model based on a mixture of word and POS tokens instead of a simple word-based language model. In this language model, content words in  $n$ -grams are replaced with their corresponding POS tags. This greatly decreases the influence of the topics of texts, as desired. It also decreases the number of parameters in the language model.

To build the language model, the following three preprocessing steps are applied to  $D_i$ . First, texts in  $D_i$  are split into sentences. Second, each sentence is tokenized, POS-tagged, and mapped entirely to lowercase. For instance, the first example sentence in Sect. 1 would give:

the/DT alien/NN would/MD not/RB  
use/VB my/PRP\$ spaceship/NN but/CC  
the/DT hers/PRP ./.

Finally, words are replaced with their corresponding POS tags; for the following words, word tokens are used as their corresponding POS tags: coordinating conjunctions, determiners, prepositions, modals, predeterminers, possessives, pronouns, question adverbs. Also, proper nouns are treated as common nouns. At this point, the special POS tags *BOS* and *EOS* are added at the beginning and end of each sentence, respectively. For instance, the above example would result in the following word/POS sequence:

BOS the NN would RB VB my NN but  
the hers . EOS

Note that the content of the original sentence is far from clear while reflecting mother tongue interference, especially in *the hers*.

Now, the language model  $M_i$  can be built from  $D_i$ . We set  $n = 3$  (i.e., trigram language model) following Kita's work and use Kneser-Ney (KN) smoothing (Kneser and Ney, 1995) to estimate its conditional probabilities.

With  $M_i$  and  $D_i$ , we can naturally apply Kita's method to our task. The clustering algorithm used is agglomerative hierarchical clustering with the average linkage method. The distance<sup>2</sup> between two language models is measured as follows. The

<sup>2</sup>It is not a distance in a mathematical sense. However, we will use the term *distance* following the convention in the literature.

probability that  $M_i$  generates  $D_i$  is calculated by  $\Pr(D_i|M_i)$ . Note that

$$\Pr(D_i|M_i) \approx \Pr(w_{1,i}) \Pr(w_{2,i}|w_{1,i}) \times \prod_{t=3}^{|D_i|} \Pr(w_{t,i}|w_{t-2,i}, w_{t-1,i}) \quad (1)$$

where  $w_{t,i}$  and  $|D_i|$  denote the  $t$ th token in  $D_i$  and the number of tokens in  $D_i$ , respectively, since we use the trigram language model. Then, the distance from  $M_i$  to  $M_j$  is defined by

$$d(M_i \rightarrow M_j) = \frac{1}{|D_j|} \log \frac{\Pr(D_j|M_j)}{\Pr(D_j|M_i)}. \quad (2)$$

In other words, the distance is determined based on the ratio of the probabilities that each language model generates the language data. Because  $d(M_i \rightarrow M_j)$  and  $d(M_j \rightarrow M_i)$  are not symmetrical, we define the distance between  $M_i$  and  $M_j$  to be their average:

$$d(M_i, M_j) = \frac{d(M_i \rightarrow M_j) + d(M_j \rightarrow M_i)}{2}. \quad (3)$$

Equation (3) is used to calculate the distance between two language models for clustering.

To sum up, the procedure of the language family tree construction method is as follows: (i) Pre-process each  $D_i$ ; (ii) Build  $M_i$  from  $D_i$ ; (iii) Calculate the distances between the language models; (iv) Cluster the language data using the distances; (v) Output the result as a language family tree.

### 3.2 Vector-based Method

We also examine a vector-based method for language family tree reconstruction. As we will see in Sect. 5, this method allows us to interpret clustering results more easily than with the language model-based method while both result in similar language family trees.

In this method,  $D_i$  is modeled by a vector. The vector is constructed based on the relative frequencies of trigrams. As a consequence, the distance is naturally defined by the Euclidean distance between two vectors. The clustering procedure is the same as for the language model-based method except that  $M_i$  is vector-based and that the distance metric is Euclidean.

## 4 Experiments

We selected the ICLE corpus v.2 (Granger et al., 2009) as the target language data. It consists of English essays written by a wide variety of non-native speakers of English. Among them, the 11 shown in Table 1 are of Indo-European languages. Accordingly, we selected the subcorpora of the 11 languages in the experiments. Before the experiments, we preprocessed the corpus data to control the experimental conditions. Because some of the writers had more than one native language, we excluded essays that did not meet the following three conditions: (i) the writer has only one native language; (ii) the writer has only one language at home; (iii) the two languages in (i) and (ii) are the same as the native language of the subcorpus to which the essay belongs<sup>3</sup>. After the selection, markup tags such as essay IDs were removed from the corpus data. Also, the symbols ‘ and ’ were unified into ’<sup>4</sup>. For reference, we also used native English (British and American university students’ essays in the LOCNESS corpus<sup>5</sup>) and two sets of Japanese English (ICLE and the NICE corpus (Sugiura et al., 2007)). Table 1 shows the statistics on the corpus data.

Performance of POS tagging is an important factor in our methods because they are based on word/POS sequences. Existing POS taggers might not perform well on non-native English texts because they are normally developed to analyze native English texts. Considering this, we tested CRFTagger<sup>6</sup> on non-native English texts containing various grammatical errors before the experiments (Nagata et al., 2011). It turned out that CRFTagger achieved an accuracy of 0.932 (compared to 0.970 on native texts). Although it did not perform as well as on native texts, it still achieved a fair accuracy. Accordingly, we decided to use it in our experiments.

Then, we generated cluster trees from the corpus data using the methods described in Sect. 3.

<sup>3</sup>For example, because of (iii), essays written by native speakers of Swedish in the Finnish subcorpus were excluded from the experiments. This is because they were collected in Finland and might be influenced by Finnish.

<sup>4</sup>The symbol ‘ is sometimes used for ’ (e.g., *I’m*).

<sup>5</sup>The LOCNESS corpus is a corpus of native English essays made up of British pupils’ essays, British university students’ essays, and American university students’ essays: <https://www.uclouvain.be/en-cecl-locness.html>

<sup>6</sup>Xuan-Hieu Phan, “CRFTagger: CRF English POS Tagger,” <http://crftagger.sourceforge.net/>, 2006.

Native language	# of essays	# of tokens
Bulgarian	294	219,551
Czech	220	205,264
Dutch	244	240,861
French	273	202,439
German	395	236,841
Italian	346	219,581
Norwegian	290	218,056
Polish	354	251,074
Russian	255	236,748
Spanish	237	211,343
Swedish	301	268,361
English	298	294,357
Japanese1 (ICLE)	171	224,534
Japanese2 (NICE)	340	130,156
Total	4,018	3,159,166

Table 1: Statistics on target corpora.

We used the Kyoto Language Modeling toolkit<sup>7</sup> to build language models from the corpus data. We removed  $n$ -grams that appeared less than five times<sup>8</sup> in each subcorpus in the language models. Similarly, we implemented the vector-based method with trigrams using the same frequency cutoff (but without smoothing).

Fig. 1 shows the experimental results. The tree at the top is the Indo-European family tree drawn based on the figure shown in Crystal (1997). It shows that the 11 languages are divided into three groups: Italic, Germanic, and Slavic branches. The second and third trees are the cluster trees generated by the language model-based and vector-based methods, respectively. The number at each branching node denotes in which step the two clusters were merged.

The experimental results strongly support the hypothesis we made in Sect. 1. Fig. 1 reveals that the language model-based method correctly groups the 11 Englishes into the Italic, Germanic, and Slavic branches. It first merges Norwegian-English and Swedish-English into a cluster. The two languages belong to the North Germanic branch of the Germanic branch and thus are closely related. Subsequently, the language model-based method correctly merges the other languages into the three branches. A dif-

<sup>7</sup>The Kyoto Language Modeling toolkit: <http://www.phontron.com/kylm/>

<sup>8</sup>We found that the results were not sensitive to the value of frequency cutoff so long as we set it to a small number.

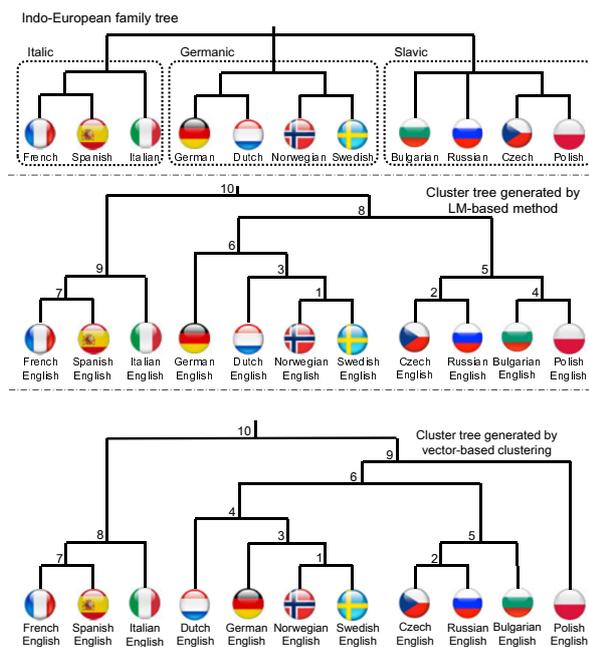


Figure 1: Experimental results.

ference between its cluster tree and the Indo-European family tree is that there are some mismatches within the Germanic and Slavic branches. While the difference exists, the method strongly distinguishes the three branches from one another. The third tree shows that the vector-based method behaves similarly while it mistakenly attaches Polish-English into an independent branch. From these results, we can say that mother tongue interference is transferred into the 11 Englishes, strongly enough for reconstructing its language family tree, which we propose calling *the interlanguage Indo-European family tree* in English.

Fig. 2 shows the experimental results with native and Japanese Englishes. It shows that the same interlanguage Indo-European family tree was reconstructed as before. More interestingly, native English was detached from the interlanguage Indo-European family tree contrary to the expectation that it would be attached to the Germanic branch because English is of course a member of the Germanic branch. This implies that non-nativeness common to the 11 Englishes is more influential than the intrafamily distance is<sup>9</sup>;

<sup>9</sup>Admittedly, we need further investigation to confirm this argument especially because we applied CRFTagger, which is developed to analyze native English, to both non-native and native Englishes, which might affect the results.

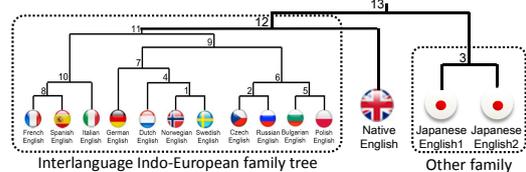


Figure 2: Experimental results with native and Japanese Englishes.

otherwise, native English would be included in the German branch. Fig. 2 also shows that the two sets of Japanese English were merged into a cluster and that it was the most distant in the whole tree. This shows that the interfamily distance is the most influential factor. Based on these results, we can further hypothesize as follows: interfamily distance > non-nativeness > intrafamily distance.

## 5 Discussion

To get a better understanding of the interlanguage Indo-European family tree, we further explore linguistic features that explain well the above phenomena. When we analyze the experimental results, however, some problems arise. It is almost impossible to find someone who has a good knowledge of the 11 languages and their mother language interference in English writing. Besides, there are a large number of language pairs to compare. Thus, we need an efficient and effective way to analyze the experimental results.

To address these problems, we did the following. First, we focused on only a few Englishes out of the 11. Because one of the authors had some knowledge of French, we selected French-English as the main target. This naturally made us select the other Italic Englishes as its counterparts. Also, because we had access to a native speaker of Russian who had a good knowledge of English, we included Russian-English in our focus. We analyzed these Englishes and then examined whether the findings obtained apply to the other Englishes or not. Second, we used a method for extracting interesting trigrams from the corpus data. The method compares three out of the 11 corpora (for example, French-, Spanish-, and Russian-Englishes). If we remove instances of a trigram from each set, the clustering tree involving

the three may change. For example, the removal of *but the hers* may result in a cluster tree merging French- and Russian-Englishes before French- and Spanish-Englishes. Even if it does not change, the distances may change in that direction. We analyzed what trigrams had contributed to the clustering results with this approach.

To formalize this approach, we will denote a trigram by  $t$ . We will also denote its relative frequency in the language data  $D_i$  by  $r_{ti}$ . Then, the change in the distances caused by the removal of  $t$  from  $D_i$ ,  $D_j$ , and  $D_k$  is quantified by

$$s = (r_{tk} - r_{ti})^2 - (r_{tj} - r_{ti})^2 \quad (4)$$

in the vector-based method. The quantity  $(r_{tk} - r_{ti})^2$  is directly related to the decrease in the distance between  $D_i$  and  $D_k$  and similarly,  $(r_{tj} - r_{ti})^2$  to that between  $D_i$  and  $D_j$  in the vector-based method. Thus, the greater  $s$  is, the higher the chance that the cluster tree changes. Therefore, we can obtain a list of interesting trigrams by sorting them according to  $s$ . We could do a similar calculation in the language model-based method using the conditional probabilities. However, it requires a more complicated calculation. Accordingly, we limit ourselves to the vector-based method in this analysis, noting that both methods generated similar cluster trees.

Table 2 shows the top 15 interesting trigrams where  $D_i$ ,  $D_j$ , and  $D_k$  are French-, Spanish-, and Russian-Englishes, respectively. Note that  $s$  is multiplied by  $10^6$  and  $r$  is in % for readability. The list reveals that many of the trigrams contain the article *a* or *the*. Interestingly, their frequencies are similar in French-English and Spanish-English, and both are higher than in Russian-English. This corresponds to the fact that French and Spanish have articles whereas Russian does not. Actually, the same argument can be made about the other Italic and Slavic Englishes (e.g., *the JJ NN*: Italian-English 0.82; Polish-English 0.72)<sup>10</sup>. An exception is that of trigrams containing the definite article in Bulgarian-English; it tends to be higher in Bulgarian-English than in the other Slavic Englishes. Surprisingly and interestingly, however, it reflects the fact that Bulgarian does have the definite article but not the indefinite article (e.g., *the JJ NN*: 0.82; *a JJ NN*: 0.60 in Bulgarian-English).

<sup>10</sup>Due to the space limitation, other lists were not included in this paper but are available at <http://web.hyogo-u.ac.jp/nagata/acl/>.

Table 3 shows that the differences in article use exist even between the Italic and Germanic branches despite the fact that both have the indefinite and definite articles. The list still contains a number of trigrams containing articles. For a better understanding of this, we looked further into the distribution of articles in the corpus data. It turns out that the distribution almost perfectly groups the 11 Englishes into the corresponding branches as shown in Fig. 3. The overall use of articles is less frequent in the Slavic-Englishes. The definite article is used more frequently in the Italic-Englishes than in the Germanic Englishes (except for Dutch-English). We speculate that this is perhaps because the Italic languages have a wider usage of the definite article such as its modification of possessive pronouns and proper nouns. The Japanese Englishes form another group (this is also true for the following findings). This corresponds to the fact that the Japanese language does not have an article system similar to that of English.

$s$	Trigram $t$	$r_{ti}$	$r_{tj}$	$r_{tk}$
5.14	the NN of	1.01	0.98	0.78
4.38	a JJ NN	0.85	0.77	0.62
2.74	the JJ NN	0.87	0.86	0.71
2.30	NN of the	0.49	0.52	0.33
1.64	. . .	0.22	0.12	0.05
1.56	NNS . EOS	0.77	0.70	0.92
1.31	NNS and NNS	0.09	0.13	0.21
1.25	BOS RB ,	0.25	0.22	0.14
1.22	of the NN	0.42	0.44	0.30
1.17	VBZ to VB	0.26	0.22	0.14
1.09	BOS i VBP	0.07	0.05	0.17
1.03	NN of NN	0.74	0.70	0.63
0.88	NN of JJ	0.15	0.15	0.25
0.67	the JJ NNS	0.28	0.28	0.20
0.65	NN to VB	0.40	0.38	0.31

Table 2: Interesting trigrams (French- ( $D_i$ ), Spanish- ( $D_j$ ), and Russian- ( $D_k$ ) Englishes).

Another interesting trigram, though not as obvious as article use, is *NN of NN*, which ranks 12th and 2nd in Table 2 and 3, respectively. In the Italic Englishes, the trigram is more frequent than the other non-native Englishes as shown in Fig. 4. This corresponds to the fact that noun-noun compounds are less common in the Italic languages than in English and that instead, the *of*-phrase (*NN of NN*) is preferred (Swan and Smith, 2001). For

$s$	Trigram $t$	$r_{ti}$	$r_{tj}$	$r_{tk}$
21.49	the NN of	1.01	0.98	0.54
5.70	NN of NN	0.74	0.70	0.50
3.26	NN of the	0.49	0.52	0.30
3.10	the JJ NN	0.87	0.86	0.70
2.62	. . .	0.22	0.12	0.03
1.53	of the NN	0.42	0.44	0.29
1.50	NN , NN	0.30	0.30	0.18
1.50	BOS i VBP	0.07	0.05	0.19
0.85	NNS and NNS	0.09	0.13	0.19
0.81	JJ NN of	0.40	0.39	0.31
0.68	. . EOS	0.13	0.06	0.02
0.63	a JJ NN	0.85	0.77	0.73
0.63	RB . EOS	0.21	0.16	0.31
0.56	NN , the	0.16	0.16	0.08
0.50	NN of a	0.17	0.09	0.06

Table 3: Interesting trigrams (French- ( $D_i$ ), Spanish- ( $D_j$ ), and Swedish- ( $D_k$ ) Englishes).

instance, *orange juice* is expressed as *juice of orange* in the Italic languages (e.g., *jus d'orange* in French). In contrast, noun-noun compounds or similar constructions are more common in Russian and Swedish. As a result, *NN of NN* becomes relatively frequent in the Italic Englishes. Fig. 4 also shows that its distribution roughly groups the 11 Englishes into the three branches. Therefore, the way noun phrases (NPs) are constructed is a clue to how the three branches were clustered.

This finding in turn reveals that the consecutive repetitions of nouns occur less in the Italic Englishes. In other words, the length tends to be shorter than in the others where we define the length as the number of consecutive repetitions of common nouns (for example, the length of *orange juice* is one because a noun is consecutively repeated once). To see if this is true, we calculated the average length for each English. Fig. 5 shows that the average length roughly distinguishes the Italic Englishes from the other non-native Englishes; French-English is the shortest, which is explained by the discussion above, while Dutch- and German-Englishes are longest, which may correspond to the fact that they have a preference for noun-noun compounds as Snyder (1996) argues. For instance, German allows the concatenated form as in *Orangensaft* (equivalently *orangejuice*). This tendency in the length of noun-noun compounds provides us with a crucial insight for native language identification, which we will

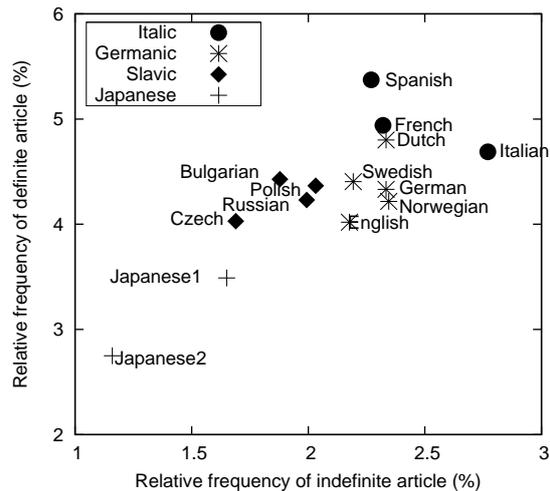


Figure 3: Distribution of articles.

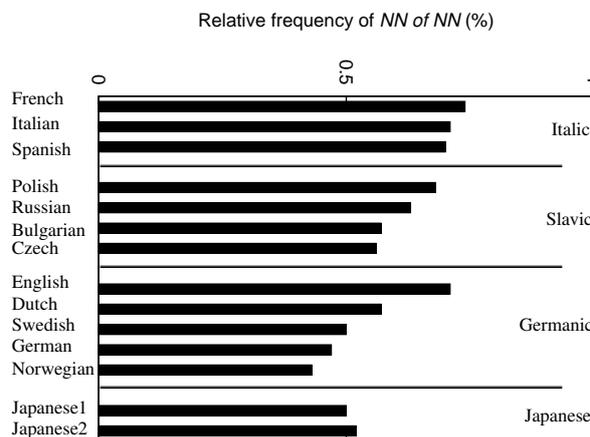


Figure 4: Relative frequency of *NN of NN* in each corpus (%).

come back to in Sect. 6.

The trigrams *BOS RB*, in Table 2 and *RB . EOS* in Table 3 imply that there might also be a certain pattern in adverb position in the 11 Englishes (they roughly correspond to adverbs at the beginning and end of sentences). Fig. 6 shows an insight into this. The horizontal and vertical axes correspond to the ratio of adverbs at the beginning and the end of sentences, respectively. It turns out that the German Englishes form a group. So do the Italic Englishes although it is less dense. In contrast, the Slavic Englishes are scattered. However, the ratios give a clue to how to distinguish Slavic Englishes from the others when combined with other

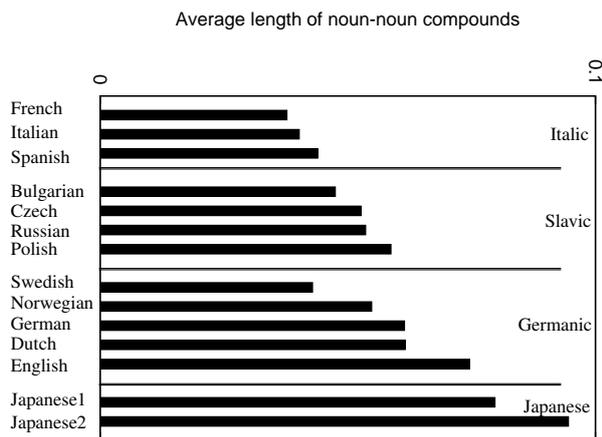


Figure 5: Average length of noun-noun compounds in each corpus.

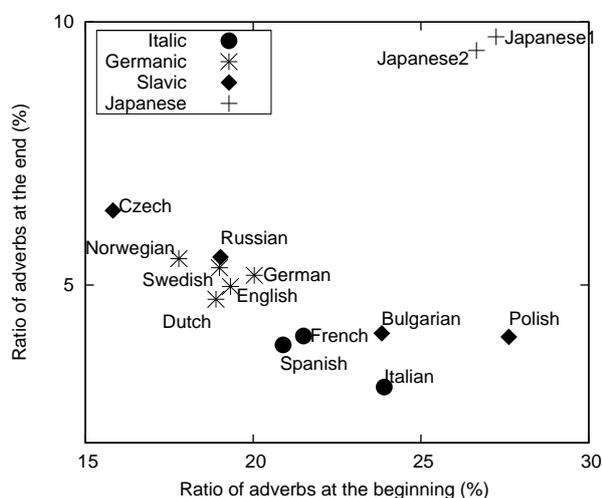


Figure 6: Distribution of adverb position.

trigrams. For instance, although Polish-English is located in the middle of Swedish-English and Bulgarian-English in the distribution of articles (in Fig. 3), the ratios tell us that Polish-English is much nearer to Bulgarian-English.

## 6 Implications for Work in Related Domains

Researchers including Wong and Dras (2009), Wong et al. (2011; 2012), and Koppel et al. (2005) work on native language identification and show that machine learning-based methods are effective. Wong and Dras (2009) propose using information about grammatical errors such as errors in determiners to achieve better performance while

they show that its use does not improve the performance, contrary to the expectation. Related to this, other researchers (Koppel and Ordan, 2011; van Halteren, 2008) show that machine learning-based methods can also predict the source language of a given translated text although it should be emphasized that it is a different task from native language identification because translation is not typically performed by non-native speakers but rather native speakers of the target language<sup>11</sup>.

The experimental results show that  $n$ -grams containing articles are predictive for identifying native languages. This indicates that they should be used in the native language identification task. Importantly, all  $n$ -grams containing articles should be used in the classifier unlike the previous methods that are based only on  $n$ -grams containing article errors. Besides, no articles should be explicitly coded in  $n$ -grams for taking the overuse/underuse of articles into consideration. We can achieve this by adding a special symbol such as  $\phi$  to the beginning of each NP whose head noun is a common noun and that has no determiner in it as in “I like  $\phi$  orange juice.”

In addition, the length of noun-noun compounds and the position of adverbs should also be considered in native language identification. In particular, the former can be modeled by the Poisson distribution as follows. The Poisson distribution gives the probability of the number of events occurring in a fixed time. In our case, the number of events in a fixed time corresponds to the number of consecutive repetitions of common nouns in NPs, which in turn corresponds to the length. To be precise, the probability of a noun-noun compound with length  $l$  is given by

$$\Pr(l) = \frac{\lambda^l}{l!} e^{-\lambda}, \quad (5)$$

where  $\lambda$  corresponds to the average length. Fig. 7 shows that the observed values in the French-English data very closely fit the theoretical proba-

<sup>11</sup>For comparison, we conducted a pilot study where we reconstructed a language family tree from English texts in European Parliament Proceedings Parallel Corpus (Europarl) (Koehn, 2011). It turned out that the reconstructed tree was different from the canonical tree (available at <http://web.hyogo-u.ac.jp/nagata/acl/>). However, we need further investigation to confirm it because each sub-corpus in Europarl is variable in many dimensions including its size and style (e.g., overuse of certain phrases such as *ladies and gentlemen*).

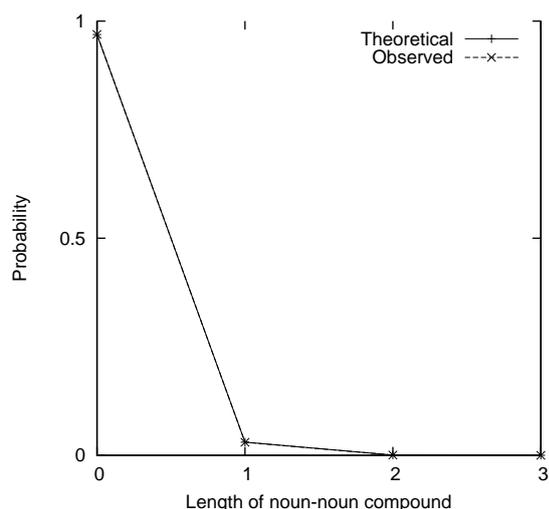


Figure 7: Distribution of noun-noun compound length for French-English.

bilities given by Equation (5)<sup>12</sup>. This holds for the other Englishes although we cannot show them because of the space limitation. Consequently, Equation (5) should be useful in native language identification. Fortunately, it can be naturally integrated into existing classifiers.

In the domain of historical linguistics, researchers have used computational and corpus-based methods for reconstructing language family trees. Some (Enright and Kondrak, 2011; Gray and Atkinson, 2003; Barbançon et al., 2007; Batagelj et al., 1992; Nakhleh et al., 2005) apply clustering techniques to the task of language family tree reconstruction. Others (Kita, 1999; Rama and Singh, 2009) use corpus statistics for the same purpose. These methods reconstruct language family trees based on linguistic features that exist within words including lexical, phonological, and morphological features.

The experimental results in this paper suggest the possibility of the use of non-native texts for reconstructing language family trees. It allows us to use linguistic features that exist between words, as seen in our methods, which has been difficult with previous methods. Language involves the features between words such as phrase construction and syntax as well as the features within words and thus they should both be considered in reconstructing

<sup>12</sup>The theoretical and observed values are so close that it is difficult to distinguish between the two lines in Fig. 7. For example,  $\Pr(l = 1) = 0.0303$  while the corresponding observed value is 0.0299.

tion of language family trees.

## 7 Conclusions

In this paper, we have shown that mother tongue interference is so strong that the relations between members of the Indo-European language family are preserved in English texts written by Indo-European language speakers. To show this, we have used clustering to reconstruct a language family tree from 11 sets of non-native English texts. It turned out that the reconstructed tree correctly groups them into the Italic, Germanic, and Slavic branches of the Indo-European family tree. Based on the resulting trees, we have then hypothesized that the following relation holds in mother tongue interference: interfamily distance  $>$  non-nativeness  $>$  intrafamily distance. We have further explored several intriguing linguistic features that play an important role in mother tongue interference: (i) article use, (ii) NP construction, and (iii) adverb position, which provide several insights for improving the tasks of native language identification and language family tree reconstruction.

## Acknowledgments

This work was partly supported by the Digiteo foreign guest project. We would like to thank the three anonymous reviewers and the following persons for their useful comments on this paper: Kotaro Funakoshi, Mitsuaki Hayase, Atsuo Kawai, Robert Ladig, Graham Neubig, Vera Sheinman, Hiroya Takamura, David Valmorin, Mikko Vilenius.

## References

- Jan Aarts and Sylviane Granger, 1998. *Tag sequences in learner corpora: a key to interlanguage grammar and discourse*, pages 132–141. Longman, New York.
- Bengt Altenberg and Marie Tapper, 1998. *The use of adverbial connectors in advanced Swedish learners' written English*, pages 80–93. Longman, New York.
- François Barbançon, Tandy Warnow, Steven N. Evans, Donald Ringe, and Luay Nakhleh. 2007. An experimental study comparing linguistic phylogenetic reconstruction methods. *Statistics Technical Reports*, page 732.
- Vladimir Batagelj, Tomaž Pisanski, and Damijana Keržič. 1992. Automatic clustering of languages. *Computational Linguistics*, 18(3):339–352.

- Robert S.P. Beekes. 2011. *Comparative Indo-European Linguistics: An Introduction (2nd ed.)*. John Benjamins Publishing Company, Amsterdam.
- Martin Chodorow, Michael Gamon, and Joel R. Tetreault. 2010. The utility of article and preposition error correction systems for English language learners: feedback and assessment. *Language Testing*, 27(3):419–436.
- David Crystal. 1997. *The Cambridge Encyclopedia of Language (2nd ed.)*. Cambridge University Press, Cambridge.
- Niels Davidsen-Nielsen and Peter Harder, 2001. *Speakers of Scandinavian languages: Danish, Norwegian, Swedish*, pages 21–36. Cambridge University Press, Cambridge.
- Alvar Ellegård. 1959. Statistical measurement of linguistic relationship. *Language*, 35(2):131–156.
- Jessica Enright and Grzegorz Kondrak. 2011. The application of chordal graphs to inferring phylogenetic trees of languages. In *Proc. of 5th International Joint Conference on Natural Language Processing*, pages 8–13.
- Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. *International Corpus of Learner English v2*. Presses universitaires de Louvain, Louvain.
- Russell D. Gray and Quentin D. Atkinson. 2003. Language-tree divergence times support the Anatolian theory of Indo-European origin. *Nature*, 426:435–438.
- Jiawei Han and Micheline Kamber. 2006. *Data Mining: Concepts and Techniques (2nd Ed.)*. Morgan Kaufmann Publishers, San Francisco.
- Bing-Hwang Juang and Lawrence R. Rabiner. 1985. A probabilistic distance measure for hidden Markov models. *AT&T Technical Journal*, 64(2):391–408.
- Kenji Kita. 1999. Automatic clustering of languages based on probabilistic models. *Journal of Quantitative Linguistics*, 6(2):167–171.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184.
- Philipp Koehn. 2011. Europarl: A parallel corpus for statistical machine translation. In *Proc. of 10th Machine Translation Summit*, pages 79–86.
- Moshe Koppel and Noam Ordan. 2011. Translationese and its dialects. In *Proc. of 49th Annual Meeting of the Association for Computational Linguistics*, pages 1318–1326.
- Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Determining an author’s native language by mining a text for errors. In *Proc. of 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 624–628.
- Alfred L. Kroeber and Charles D. Chriétien. 1937. Quantitative classification of Indo-European languages. *Language*, 13(2):83–103.
- Ryo Nagata, Edward Whittaker, and Vera Sheinman. 2011. Creating a manually error-tagged and shallow-parsed learner corpus. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1210–1219.
- Luay Nakhleh, Tandy Warnow, Don Ringe, and Steven N. Evans. 2005. A comparison of phylogenetic reconstruction methods on an Indo-European dataset. *Transactions of the Philological Society*, 103(2):171–192.
- Taraka Rama and Anil Kumar Singh. 2009. From bag of languages to family trees from noisy corpus. In *Proc. of Recent Advances in Natural Language Processing*, pages 355–359.
- Anna Giacalone Ramat and Paolo Ramat, 2006. *The Indo-European Languages*. Routledge, New York.
- William Snyder. 1996. The acquisitional role of the syntax-morphology interface: Morphological compounds and syntactic complex predicates. In *Proc. of Annual Boston University Conference on Language Development*, volume 2, pages 728–735.
- Masatoshi Sugiura, Masumi Narita, Tomomi Ishida, Tatsuya Sakaue, Remi Murao, and Kyoko Muraki. 2007. A discriminant analysis of non-native speakers and native speakers of English. In *Proc. of Corpus Linguistics Conference CL2007*, pages 84–89.
- Michael Swan and Bernard Smith. 2001. *Learner English (2nd Ed.)*. Cambridge University Press, Cambridge.
- Hans van Halteren. 2008. Source language markers in EUROPARL translations. In *Proc. of 22nd International Conference on Computational Linguistics*, pages 937–944.
- Sze-Meng J. Wong and Mark Dras. 2009. Contrastive analysis and native language identification. In *Proc. Australasian Language Technology Workshop*, pages 53–61.
- Sze-Meng J. Wong, Mark Dras, and Mark Johnson. 2011. Exploiting parse structures for native language identification. In *Proc. Conference on Empirical Methods in Natural Language Processing*, pages 1600–1611.
- Sze-Meng J. Wong, Mark Dras, and Mark Johnson. 2012. Exploring adaptor grammars for native language identification. In *Proc. Joint Conference on*

*Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 699–709.

# Word Association Profiles and their Use for Automated Scoring of Essays

Beata Beigman Klebanov and Michael Flor

Educational Testing Service

660 Rosedale Road

Princeton, NJ 08541

{bbeigmanklebanov,mflor}@ets.org

## Abstract

We describe a new representation of the content vocabulary of a text we call *word association profile* that captures the proportions of highly associated, mildly associated, unassociated, and dis-associated pairs of words that co-exist in the given text. We illustrate the shape of the distribution and observe variation with genre and target audience. We present a study of the relationship between quality of writing and word association profiles. For a set of essays written by college graduates on a number of general topics, we show that the higher scoring essays tend to have higher percentages of both highly associated and dis-associated pairs, and lower percentages of mildly associated pairs of words. Finally, we use word association profiles to improve a system for automated scoring of essays.

## 1 Introduction

The vast majority of contemporary research that investigates statistical properties of language deals with characterizing *words* by extracting information about their behavior from large corpora. Thus, co-occurrence of words in  $n$ -word windows, syntactic structures, sentences, paragraphs, and even whole documents is captured in vector-space models built from text corpora (Turney and Pantel, 2010; Basili and Pennacchiotti, 2010; Erk and Padó, 2008; Mitchell and Lapata, 2008; Bullinaria and Levy, 2007; Jones and Mewhort, 2007; Pado and Lapata, 2007; Lin, 1998; Landauer and Dumais, 1997; Lund and Burgess, 1996; Salton et al., 1975). However, little is known about typical profiles of *texts* in terms of co-occurrence behavior of their words. Some information can be inferred from the success of statistical techniques in predicting certain structures in text. For example, the

fact that a text segmentation algorithm that uses information about patterns of word co-occurrences can detect sub-topic shifts in a text (Riedl and Bie-mann, 2012; Misra et al., 2009; Eisenstein and Barzilay, 2008) tells us that texts contain some proportion of more highly associated word pairs (those in subsequent sentences within the same topical unit) and of less highly associated pairs (those in sentences from different topical units).<sup>1</sup> Yet, does each text have a different distribution of highly associated, mildly associated, unassociated, and dis-associated pairs of words, or do texts tend to strike a similar balance of these? What are the proportions of the different levels of association, how much variation there exists, and are there systematic differences between various kinds of texts? We present research that makes a first step in addressing these questions.

From the applied perspective, our interest is in quantifying differences between well-written and poorly written essays, for the purposes of automated scoring of essays. We therefore concentrate on essay data for the main experiments reported in this paper, although some additional corpora will be used for illustration purposes.

The paper is organized as follows. Section 2 presents our methodology for building word association profiles for texts. Section 3 illustrates the profiles for three corpora from different genres. Section 4.2 presents our study of the relationship between writing quality and patterns of word associations, with section 4.5 showing the results of adding a feature based on word association profile to a state-of-art essay scoring system. Related work is reviewed in section 5.

---

<sup>1</sup>Note that the classical approach to topical segmentation of texts, TextTiling (Hearst, 1997), uses only word repetitions. The cited approaches use topic models that are in turn estimated using word co-occurrence.

## 2 Methodology

In order to describe the word association profile of a text, three decisions need to be made. The first decision is how to quantify the extent of co-occurrence between two words; we will use point-wise mutual information (**PMI**) estimated from a large and diverse corpus of texts. The second is which pairs of words in a text to consider when building a profile for the text; we opted for all pairs of content word types occurring in a text, irrespective of the distance between them. We consider word types, not tokens; no lemmatization is performed. The third decision is how to represent the co-occurrence profiles; we use a histogram where each bin represents the proportion of word pairs in the given interval of PMI values. The rest of the section gives more detail about these decisions.

To obtain comprehensive information about typical co-occurrence behavior of words of English, we build a first-order co-occurrence word-space model (Turney and Pantel, 2010; Baroni and Lenci, 2010). The model was generated from a corpus of texts of about 2.5 billion words, counting co-occurrence in a paragraph,<sup>2</sup> using no distance coefficients (Bullinaria and Levy, 2007). About 2 billion words come from the Gigaword 2003 corpus (Graff and Cieri, 2003). Additional 500 million words come from an in-house corpus containing popular science and fiction texts. Occurrence counts of 2.1 million word types and of 1,279 million word type pairs are efficiently compressed using the TrendStream technology (Flor, 2013), resulting in a database file of 4.7GB. TrendStream is a trie-based architecture for storage, retrieval, and updating of very large word n-gram datasets. We store pairwise word associations as bigrams; since associations are unordered, only one of the orders is actually stored in the database.

There is an extensive literature on the use of word-association measures for NLP, especially for detection of collocations (Pecina, 2010; Evert, 2008; Futagi et al., 2008). The use of point-wise mutual information with word-space models is noted in (Zhang et al., 2012; Baroni and Lenci, 2010; Mitchell and Lapata, 2008; Turney, 2001). Point-wise mutual information is defined as follows (Church and Hanks, 1990):

<sup>2</sup>In all texts, we use human-marked paragraphs, indicated either by a new line or by an xml markup.

$$PMI(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)} \quad (1)$$

Differently from Church and Hanks (1990), we disregard word order when computing  $P(x, y)$ . All probabilities are estimated using frequencies.

We define **WAP<sub>T</sub>** – a **word association profile** of a text  $T$  – as the distribution of  $PMI(x, y)$  for all pairs of content<sup>3</sup> word types  $(x, y) \in T$ . All pairs of word types for which the associations database returned a null value (the pair has never been observed in the same paragraph) are excluded from the calculation. For our main dataset (described later as setA, section 4.1), the average percentage of non-null values per text is 92%.

To represent the WAP of a text, we use a 60-bin histogram spanning all PMI values. The lowest bin (shown in Figures 1 and 2 as  $PMI = -5$ ) contains pairs with  $PMI \leq -5$ ; the topmost bin (shown in Figures 1 and 2 as  $PMI = 4.83$ ) contains pairs with  $PMI > 4.67$ , while the rest of the bins contain word pairs  $(x, y)$  with  $-5 < PMI(x, y) \leq 4.67$ . Each bin in the histogram (apart from the top and the bottom ones) corresponds to a PMI interval of 0.167. We chose a relatively fine-grained binning and performed no optimization for grid selection; for more sophisticated gridding approaches to study non-linear relationships in the data, see Reshef et al. (2011).

We will say that a text A is **tighter** than text B if the WAP of A is shifted towards the higher end of PMI values relative to text B. The intuition behind the terminology is that texts with higher proportions of highly associated pairs are likelier to be more focused, dealing with a small number of topics at greater length, as opposed to texts that bring various different themes into the text to various extents. Thus, the text “The dog barked and wagged its tail” is much tighter than the text “Green ideas sleep furiously”, with all the six content word pairs scoring above  $PMI=5.5$  in the first and below  $PMI=2.2$  in the second.<sup>4</sup>

## 3 Illustration: The shape of the distribution

For a first illustration, we use a corpus of 5,904 essays written as part of a standardized graduate

<sup>3</sup>We part-of-speech tag a text using OpenNLP tagger (<http://opennlp.apache.org>) and only take into account common and proper nouns, verbs, adjectives, and adverbs.

<sup>4</sup>We omitted *colorless* from the second example, as *colorless* is actually highly associated with *green* ( $PMI=4.36$ ).

school admission test (a full description of these data is given in section 4.1, under setA p1-p6). For each essay, we compute the WAP and represent it using the 60-bin histogram. For each bin in the histogram, we compute its average value over the 5,904 essays; additionally, we compute the 15<sup>th</sup> and 85<sup>th</sup> percentiles for each bin, so that the band between them contains values observed for 70% of the texts. The series with the solid thick (blue) line in Figure 1 shows the distribution of the average percentage of word type pairs per bin (essays-av); the dotted lines above and below show the band capturing the middle 70% of the distribution (essays-15 and essays-85).

We observe that the shape of the WAP is very stable across essays, and the variation around the average is quite limited.

Next, consider the thin solid (green) line with asterisk-shaped markers in Figure 1 that plots a similarly-binned histogram for the normal distribution with  $\mu=0.90$  and  $\sigma=0.66$ . We note that for values below  $\text{PMI}=2.17$ , the normal curve is within or almost within the 70% band for the essay data. The divergence occurs at the right tail with  $\text{PMI}>2.17$ , that covers, on average, about 8% of the pairs (5.6% and 10.4% for the 15<sup>th</sup> and 85<sup>th</sup> percentiles, respectively).

To get an idea about possible variation in the distribution, we consider two additional corpora from different genres. We use a corpus of Wall Street Journal 1987 articles from the TIPSTER collection.<sup>5</sup> We picked articles of 250 to 700 words in length, in order to keep the length of texts comparable to the essay data, while varying the genre; 770 such articles were found. The dashed (orange) line in Figure 1 shows the distribution of average values for the WSJ collection (wsj-av). We observe that the shape of the distribution is similar to that of essay data, although WSJ articles tend to be less tight, on average, since the distribution in  $\text{PMI}<2.17$  area in the WSJ data is shifted to the left relative to essays. Yet, the picture at the right tail is remarkably similar to that of the essays, with 9% of word pairs, on average, having  $\text{PMI}>2.17$ .

The second additional corpus contains 140 literary texts written or adapted for readers in grades 3 and 4 in US schools (Sheehan et al., 2008). In terms of length, these texts fall into the same range as the other corpora, averaging 507 words.

<sup>5</sup>LDC93T3A in LDC catalogue

The average WAP for these texts is shown with a thin solid (purple) line with circular markers in Figure 1 (Grades 3-4). These texts are much tighter than texts in the other two collections, as the distribution is shifted to the right. The right tail, with  $\text{PMI}>2.17$ , holds 19% of all word pairs in these texts – more than twice the proportion in essays written by college graduates or in texts from the WSJ.

It is instructive to check whether the over-use of highly associated pairs is *felt* during reading. These texts strike an adult reader as overly explicit, taking the space to state things that an adult reader would readily infer or assume. For example, consider the following opening paragraph:

“Grandma Rose gave Daniel a recorder. A recorder is a musical instrument. Daniel learned to play by blowing on the recorder. It didn’t take lots of air. It didn’t take big hands to hold since it was pocket-sized. His fingers covered the toneholes just fine. Soon Daniel played entire songs. His mother loved to listen. Sometimes she hummed along with Daniel’s recorder.”

The second and the third sentences state things that for an adult reader would be too obvious to need mention. In fact, these sentences almost seem like *training* sentences – the kind of sentences from which the associations between *recorder* and *musical instrument*, *play*, *blowing* can be learned. According to Hoey’s theory of lexical priming (Hoey, 2005), one of the main functions of schooling is to imbue children with the societally sanctioned word associations.

To conclude the illustration, we observe that there are some broad similarities between the different corpora in terms of the distribution of pairs of word types. Thus, texts seem to be mainly made of pairs of weakly associated words – about half the pairs of word types lie between PMI of 0.5 and 1.5, in all the examined collections (52% for essays, 44% for each of WSJ and young reader corpora). The percentages of pairs at the low and the high ends of PMI differ with genre – writing for children favors the higher end, while typical Wall Street Journal writing favors the low end, relatively to a corpus of essays on general topics written by college graduates.

These observations are necessarily very tentative, as only a few corpora were examined. Still,

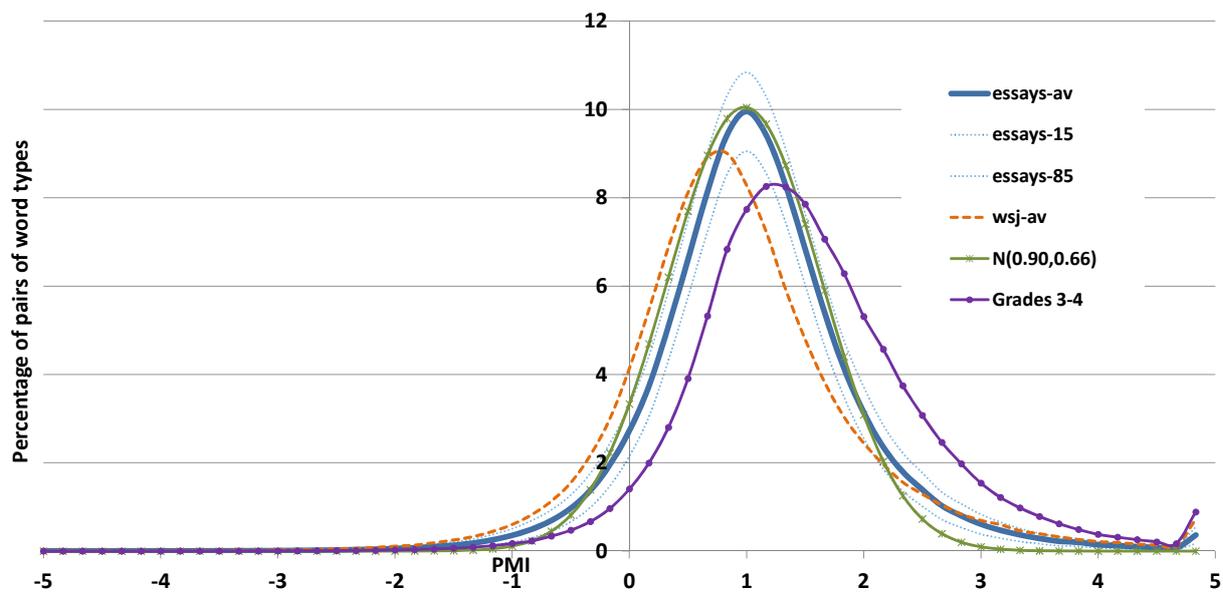


Figure 1: WAP histograms for three corpora, shown with smooth lines instead of bars for readability. Average for essays (a thick solid blue line), average for WSJ articles (a dashed orange line); average for Grades 3-4 corpus (a thin solid purple line with round markers). Normal distribution is shown with a thin solid green line with asterisk markers. Middle 70% of essays fall between the dotted lines.

we believe the illustration is suggestive, in that there is both constancy in writing for a similar purpose (observe the limited variation around the average that captures 70% of the essays) and variation with genre and target audience. In what follows, we will explore more thoroughly the information provided by word association profiles regarding the quality of writing.

#### 4 Application to Essay Scoring

Texts written for a test and scored by relevant professionals is a setting where variation in text quality is expected. In this section, we report our experiments with using WAPs to explore the variation in quality as quantified by essay scores. We first describe the data (section 4.1), then show the patterns of relationships between essay scores and word association profiles (section 4.2). Finally, we report on an experiment where we significantly improve the performance of a very competitive, state-of-art system for automated scoring of essays, using a feature derived from WAP.

##### 4.1 Data

We consider two collections of essays written as responses in an analytical writing section of a high-stakes standardized test for graduate school admission; the time limit for essay composition was 45 minutes. Essays were written in response

to a prompt (essay question). A prompt is usually a general statement, and the test-taker is asked to develop an argument supporting or refuting the statement. Example prompts are: “High-speed electronic communications media, such as electronic mail and television, tend to prevent meaningful and thoughtful communication” and “In the age of television, reading books is not as important as it once was. People can learn as much by watching television as they can by reading books.”

The first collection (henceforth, **setA**) contains 8,899 essays written in response to nine different prompts, about 1,000 per prompt;<sup>6</sup> the per-prompt subsets will be termed **setA-p1** through **setA-p9**. Each essay in setA was scored by 1 to 4 human raters on a scale of 1 to 6; the majority of essays received 2 human scores. We use the average of the available human scores as the gold-standard score for the essay. Most essays thereby receive an integer score,<sup>7</sup> so the ranking of the essays is coarse. From this set, p1-p6 were used for feature selection, data visualization, and estimation of the regression models (training), while sets p7-p9 were reserved for a blind test.

The second collection (henceforth, **setB**) con-

<sup>6</sup>While we sampled exactly 1,000 essays per prompt, we removed empty responses, resulting in 975 to 1,000 essays per sample.

<sup>7</sup>as the two raters agree most of the time

tains 400 essays, with 200 essays written on each of two prompts given as examples above (**setB-p1** and **setB-p2**). In an experimental study by Attali et al. (2013), each essay was scored by 16 professional raters on a scale of 1 to 6, allowing plus and minus scores as well, quantified as 0.33 – thus, a score of 4- is rendered as 3.67. This fine-grained scale resulted in higher mean pairwise inter-rater correlations than the traditional integer-only scale ( $r=0.79$  vs around  $r=0.70$  for the operational scoring). We use the average of 16 raters as the final grade for each essay. This dataset provides a very fine-grained ranking of the essays, with almost no two essays getting exactly the same score.

Rounded Score	setA p1-p9			setB	
	av	min	max	p1	p2
1	.01	.00	.01	–	–
2	.05	.04	.06	.03	.03
3	.25	.20	.29	.30	.28
4	.44	.42	.47	.54	.55
5	.21	.16	.24	.13	.14
6	.04	.02	.07	.01	.02

Table 1: Score distribution in the essay data. For the sake of presentation in this table, all scores were rounded to integer scores, so a score of 3.33 was counted as 3, and a score of 3.5 was counted as 4. A cell with the value of .13 (row titled 5 and column titled SetB p1) means that 13% of the essays in setB-p1 received scores that round to 5. For setA, average, minimum, and maximum values across the nine prompts are shown.

Table 1 shows the distribution of rounded scores in both collections. Average essay scores are between 3.74 to 3.98 across the different prompts from both collections. The use of 16 raters seems to have moved the rounded scores towards the middle; however, the relative ranking of the essays is much more delicate in setB than in setA.

## 4.2 Essay Score vs WAP

We calculated correlations between essay score and the proportion of word pairs in each of the 60 bins of the WAP histogram, separately for each of the prompts p1-p6 in setA. For a sample of 1,000 instances, a correlation of  $r=0.065$  is significant at  $p = 0.05$ . Figure 2 plots the correlations.

First, we observe that, perhaps contrary to expectation, the proportion of the highest values of PMI (the area to the right of  $PMI=4$  in Figure 2)

does not yield a consistent correlation with essay scores. Thus, inasmuch as highest PMI values tend to capture multi-word expressions (*South* and *Africa*; *Merill* and *Lynch*), morphological variants (*bids* and *bidding*), or synonyms (*mergers* and *takeovers*), their proportion in word type pairs does not seem to give a clear signal regarding the quality of writing.<sup>8</sup>

In contrast, the area of moderately high PMI values (from  $PMI=2.5$  to  $PMI=3.67$  in Figure 2) produces a very consistent picture, with only two points out of 48 in that interval<sup>9</sup> lacking significant positive correlation with essay score (p2 at  $PMI=3.17$  and p5 at  $PMI=3$ ).

Next, observe the consistent negative correlations between essay score and the proportion of word pairs in bins  $PMI=0.833$  through  $PMI=1.5$ . Here again, out of the 30 data points corresponding to these values, only 3 failed to reach statistical significance, although the trend there is still negative.

Finally, there is a trend towards a positive correlation between essay scores and the proportion of mildly negative PMI values ( $-2 < PMI < 0$ ), that is, better essays tend to use *more* pairs of disassociated words, although this trend is not as clear-cut as the one on the right-hand side of the distribution.

Assuming that a higher proportion of high PMI pairs corresponds to more topic development and that a higher proportion of negative PMIs corresponds to more creative use of language (in that pairs are chosen that do not generally tend to appear together), it seems that the better essays are *both* more topical *and* more creative than the lower scoring ones. In what follows, we check whether the information about essay quality provided by WAP can be used to improve essay scoring.

<sup>8</sup>It is also possible that some of the instances with very high PMI are pairs that contain low frequency words for which the database predicts a spuriously high PMI based on a single (and a-typical) co-occurrence that happens to repeat in an essay – similar to the *Schwartz eschews* example in (Manning and Schütze, 1999, Table 5.16, p. 181). On the one hand, we do not expect such pairs to occur in any systematic pattern, so they could obscure an otherwise more systematic pattern in the high PMI bins. On the other hand, we do not expect to see many such pairs, simply because a repetition of an a-typical event is likely to be very rare. We thank an anonymous reviewer for suggesting this direction, and leave a more detailed examination of the pairs in the highest-PMI bins to future work.

<sup>9</sup>There are 8 bins of width of 0.167 in the given interval, with 6 datapoints per bin.

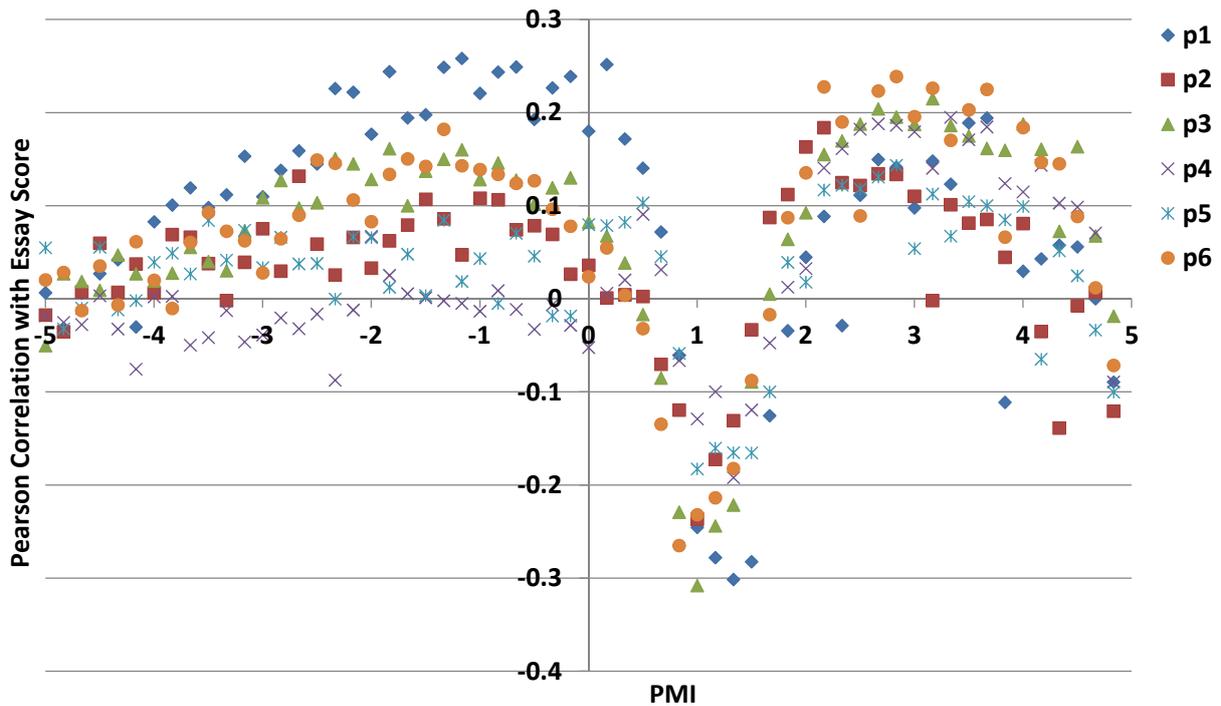


Figure 2: Correlations with essay score for various bins of the WAP histogram. P1 to P6 correspond to the first 6 prompts in SetA.

### 4.3 Baseline

As a baseline, we use e-rater (Attali and Burstein, 2006), a state-of-art essay scoring system developed at Educational Testing Service.<sup>10</sup> E-rater computes more than 100 micro-features, which are aggregated into macro-features aligned with specific aspects of the writing construct. The system incorporates macro-features measuring grammar, usage, mechanics, style, organization and development, lexical complexity, and vocabulary usage. Table 2 gives examples of micro-features covered by the different macro-features.

E-rater models are built using linear regression on large samples of test-taker essays. We use a generic e-rater model built at Educational Testing Service using essays across a variety of writing prompts, with no connection to the current project and its authors. This model obtains Pearson correlations of  $r=0.8324-0.8721$  with the human scores on setA, and the staggering  $r=0.9191$  and  $r=0.9146$  with the human scores on setB-p1 and setB-p2, respectively. This is a very competitive baseline, as e-rater features explain more than 70% of the variation in essay scores on a relatively coarse scale (setA) and more than 80% of the variation in scores on a fine-grained scale (setB).

<sup>10</sup><http://www.ets.org/erater/about/>

Macro-Feature	Example Micro-Features
Grammar, Usage, and Mechanics	agreement errors verb formation errors missing punctuation
Style	passive very long or short sentences excessive repetition
Organization and Development	use of discourse elements: thesis, support, conclusion
Lexical Complexity	average word frequency average word length
Vocabulary	similarity to vocabulary in high- vs low-scoring essays

Table 2: Features used in e-rater (Attali and Burstein, 2006).

### 4.4 Adding WAP

We define **HAT** – high associative tightness – as the percentage of word type pairs with  $2.33 < PMI \leq 3.67$  (bins  $PMI=2.5$  through  $PMI=3.67$ ). This range corresponds to the longest sequence of adjacent bins in the  $PMI > 0$  area that had a positive correlation with essay score in the setA-p1 set. The HAT feature attains significant

(at  $p = 0.05$ ) correlations with essay scores,  $r=0.11$  to  $r=0.27$  for the prompts in setA, and  $r=0.22$  and  $r=0.21$  for the two prompts in setB. We note that the HAT feature is not correlated with essay length. Essay length is not used as a feature in e-rater models, but it typically correlates strongly with the human essay score (at about  $r=0.70$  in our data), as well as with the score provided by e-rater (at about  $r=0.80$ ).

We also explored a feature that captured the area with the negative correlations identified in section 4.2. This feature did not succeed in improving the performance over the baseline on setA p1-p6; we tentatively conclude that information contained in that feature, i.e. the proportion of mildly associated vocabulary in an essay, is indirectly captured by another feature or group of features already present in e-rater. Likewise, a feature that calculates the average PMI for all pairs of content word types in the text failed to produce an improvement over the baseline for setA p1-p6. The reason for this can be observed in Figure 2: The higher-scoring essays having more of *both* the low and the high PMI pairs leads to about the same average PMI as for the lower-scoring essays that have a higher concentration of values closer to the average PMI.

#### 4.5 Evaluation

To evaluate the usefulness of WAP in improving automated scoring of essays, we estimate a linear regression model using the human score as a dependent variable (label) and e-rater score and the HAT as the two independent variables (features). The correlations between the two independent variables (e-rater and HAT) are between  $r=0.11$  and  $r=0.24$  on the prompts in setA and setB.

We estimate a regression model on each of setA- $p_i$ ,  $i \in \{1, \dots, 6\}$ , and evaluate them on each of setA- $p_j$ ,  $j \in \{7, \dots, 9\}$ , and compare the performance with that of e-rater alone on setA- $p_j$ . Note that e-rater itself is not trained on any of the data in setA and setB; we use the same e-rater model for all evaluations, a generic model that was pre-trained on a large number of essays across different prompts. For setB, we estimate the regression model on setB-p1 and test on setB-p2, and vice versa.

Table 3 shows the evaluation results. The HAT feature leads to a statistically significant improve-

Train	Test	E-rater on Test	E-rater+HAT on Test	$t$
<b>setA</b>				
p1	p7	0.84043	0.84021	-0.371
p2	p7	0.84043	0.84045	0.408
p3	p7	0.84043	0.83999	-0.597
p4	p7	0.84043	0.84044	0.411
p5	p7	0.84043	0.84028	-0.280
p6	p7	0.84043	0.83926	-1.080
p1	p8	0.83244	0.83316	<u>1.688</u>
p2	p8	0.83244	0.83250	<u>2.234</u>
p3	p8	0.83244	0.83327	1.530
p4	p8	0.83244	0.83250	<u>2.237</u>
p5	p8	0.83244	0.83311	<u>1.752</u>
p6	p8	0.83244	0.83339	1.191
p1	p9	0.86370	0.86612	<u>4.282</u>
p2	p9	0.86370	0.86389	<u>5.205</u>
p3	p9	0.86370	0.86659	<u>4.016</u>
p4	p9	0.86370	0.86388	<u>5.209</u>
p5	p9	0.86370	0.86591	<u>4.390</u>
p6	p9	0.86370	0.86730	<u>3.448</u>
<b>setB</b>				
p1	p2	0.9146	0.9178	0.983
p2	p1	0.9191	0.9242	<u>2.690</u>

Table 3: Performance of baseline model (e-rater) and models where e-rater was augmented with HAT, a feature based on the word association profile. Performance is measured using Pearson correlation with essay score. We use Wilcoxon Signed-Ranked test for matched pairs, and report the sum of signed ranks (W), the number of ranks (n), and the  $p$  value. E-rater+HAT is significantly better than e-rater alone,  $W=138$ ,  $n=20$ ,  $p<0.05$ . We also measure significance of the improvement for each row individually, using McNemar’s test for significance of difference in same-sample correlations (McNemar, 1955, p.148); we report the  $t$  value for each test. For values of  $t > 1.645$ , we can reject the hypothesis that e-rater+HAT is not better than e-rater alone with 95% confidence. Significant improvements are underlined.

ment in the performance of automated scoring. An improvement is observed for 14 out of the 18 evaluations for setA, as well as for both evaluations for setB.<sup>11</sup> Moreover, the largest relative improvement of 0.55%, from 0.9191 to 0.9242, was observed for the setting with the highest baseline performance, suggesting that the HAT feature is still effective even after the delicate ranking of the essays revealed an exceptionally strong performance of e-rater.

## 5 Related Work

Most of the attention in the computational linguistics research that deals with analysis of the lexis of texts has so far been paid to what in our terms would be the very high end of the word association profile. Thus, following Halliday and Hasan (1976), Hoey (1991), and Morris and Hirst (1991), the notion of *lexical cohesion* has been used to capture repetitions of words and occurrence of words with related meanings in a text. Lexically cohesive words are traced through the text, forming lexical chains or graphs, and these representations are used in a variety of applications, such as segmentation, keyword extraction, summarization, sentiment analysis, temporal indexing, hypelink generation, error correction (Guinaudeau et al., 2012; Marathe and Hirst, 2010; Ercan and Cicekli, 2007; Devitt and Ahmad, 2007; Hirst and Budanitsky, 2005; Inkpen and Désilets, 2005; Gurevych and Strube, 2004; Stokes et al., 2004; Silber and McCoy, 2002; Green, 1998; Al-Halimi and Kazman, 1998; Barzilay and Elhadad, 1997). To our knowledge, lexical cohesion has not so far been used for automated scoring of essays. Our results suggest that this direction is promising, as merely the *proportion* of highly associated word pairs is already contributing a clear signal regarding essay quality; it is possible that additional information can be derived from richer representations common in the lexical cohesion literature.

Aspects related to the distribution of words in essays have been studied in relation to essay scoring. One line of work focuses on assessing coherence of essays. Foltz et al. (1998) use Latent

<sup>11</sup>We also performed a cross-validation test on setA p1-p6, where we estimated a regression model on setA-p*i* and evaluate it on setA-p*j*, for all  $i, j \in \{1, \dots, 6\}, i \neq j$ , and compared the performance with that of e-rater alone on setA-p*j*, yielding 30 different train-test combinations. The results were similar to those of the blind test presented here, with e-rater+HAT significantly improving upon e-rater alone, using Wilcoxon test,  $W=374, n=29, p<0.05$ .

Semantic Analysis to model the smoothness of transitions between adjacent segments of an essay. Higgins et al. (2004) compare sentences from certain discourse segments in an essay to determine their semantic similarity, such as comparing thesis statements to conclusions or thesis statements to essay prompts. Additional approaches include evaluation of coherence based on repeated reference to entities (Burstein et al., 2010; Barzilay and Lapata, 2008; Miltsakaki and Kukich, 2004). Our approach is different in that it does not measure the flow of the text, that is, the sequencing and repetition of the words, but rather assesses the choice of vocabulary as a whole.

Topic models have been proposed as a technique for capturing clusters of related words that tend to occur in the same documents in a given collection. A text is modeled as being composed of a small number of topics, and words in the text are generated conditioned on the selected topics (Gruber et al., 2007; Blei et al., 2003). Since (a) topics encapsulate clusters of highly associated words, and (b) topics for a given text are modeled as being chosen independently from each other, we expect a negative correlation between the number of topics in a document and the tightness of the word association profile of the text.

An alternative representation of word association profile would be a weighted graph, where the weights correspond to pairwise associations between words. Thus, for longer texts, graph analysis techniques would be applicable. Steyvers and Tenenbaum (2005) analyze the graphs induced from large repositories like WordNet or databases of free associations, and find them to be scale-free and small-world; it is an open question whether word association graphs induced from book-length texts would exhibit similar properties.

In the theoretical tradition, our work is closest in spirit to Michael Hoey's theory of lexical priming (Hoey, 2005), positing that users of language internalize patterns of occurrence and non-occurrence of words not only with other words, but also in certain positions in a text, in certain syntactic environments, and in certain evaluative contexts, and use these when creating their own texts. We believe that word association profiles reflect the artwork that goes into using those internalized associations between words when creating a text, achieving the right mix of strong and weak, positive and negative associations.

## 6 Conclusion

In this paper, we described a new representation of the content vocabulary of a text we call *word association profile* that captures the proportions of highly associated, mildly associated, unassociated, and dis-associated pairs of words selected to co-exist in the given text by its author. We observed that the shape of the distribution is quite stable across various texts, with about half the pairs having a mild association; the allocation of pairs to the higher and the lower levels of association does vary across genres and target audiences.

We further presented a study of the relationship between quality of writing and word association profiles. For a dataset of essays written by college graduates on a number of general topics in a standardized test for graduate school admission and scored by professional raters, we showed that the higher scoring essays tend to have higher percentages of both highly associated and dis-associated pairs, and lower percentages of mildly associated pairs of words. We hypothesize that this pattern is consistent with the better essays demonstrating both a better topic development (hence the higher percentage of highly related pairs) and a more creative use of language resources, as manifested in a higher percentage of word pairs that generally do not tend to appear together.

Finally, we demonstrated that the information provided by word association profiles leads to a significant improvement in a highly competitive, state-of-art essay scoring system that already measures various aspects of writing quality.

In future work, we intend to investigate in more detail the contribution of various kinds of words to word association profiles, as well as pursue application to evaluation of text complexity.

## References

Reem Al-Halimi and Rick Kazman. 1998. Temporal indexing through lexical chaining. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 333–351. Cambridge, MA: MIT Press.

Yigal Attali and Jill Burstein. 2006. Automated Essay Scoring With e-rater®V.2. *Journal of Technology, Learning, and Assessment*, 4(3).

Yigal Attali, Will Lewis, and Michael Steier. 2013. Scoring with the computer: Alternative procedures for improving reliability of holistic essay scoring. *Language Testing*, 30(1):125–141.

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

Regina Barzilay and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of ACL Intelligent Scalable Text Summarization Workshop*.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Roberto Basili and Marco Pennacchiotti. 2010. Distributional lexical semantics: Toward uniform representation paradigms for advanced acquisition and processing tasks. *Natural Language Engineering*, 16(4):347–358.

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

John Bullinaria and Joseph Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39:510–526.

Jill Burstein, Joel Tetreault, and Slava Andreyev. 2010. Using entity-based features to model coherence in student essays. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 681–684, Los Angeles, California, June. Association for Computational Linguistics.

Kenneth Church and Patrick Hanks. 1990. Word association norms, mutual information and lexicography. *Computational Linguistics*, 16(1):22–29.

Ann Devitt and Khurshid Ahmad. 2007. Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 984–991, Prague, Czech Republic, June. Association for Computational Linguistics.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 334–343, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gonenc Ercan and Ilyas Cicekli. 2007. Using lexical chains for keyword extraction. *Information Processing & Management*, 43(6):1705–1714.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 897–906, Honolulu, Hawaii, October. Association for Computational Linguistics.

- Stefan Evert. 2008. Corpora and collocations. In A. Lüdeling and M. Kytö, editors, *Corpus Linguistics: An International Handbook*. Berlin: Mouton de Gruyter.
- Michael Flor. 2013. A fast and flexible architecture for very large word n-gram datasets. *Natural Language Engineering*, 19(1):61–93.
- Peter Foltz, Walter Kintsch, and Thomas Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25(2):285–307.
- Yoko Futagi, Paul Deane, Martin Chodorow, and Joel Tetreault. 2008. A computational approach to detecting collocation errors in the writing of non-native speakers of English. *Computer Assisted Language Learning*, 21(4):353–367.
- David Graff and Christopher Cieri. 2003. English Gigaword LDC2003T05. Linguistic Data Consortium, Philadelphia.
- Stephen Green. 1998. Automated link generation: Can we do better than term repetition? *Computer Networks*, 30:75–84.
- Amit Gruber, Yair Weiss, and Michal Rosen-Zvi. 2007. Hidden topic markov models. *Journal of Machine Learning Research - Proceedings Track*, 2:163–170.
- Camille Guinaudeau, Guillaume Gravier, and Pascale Sébillot. 2012. Enhancing lexical cohesion measure with confidence measures, semantic relations and language model interpolation for multimedia spoken content topic segmentation. *Computer Speech and Language*, 26(2):90–104.
- Iryna Gurevych and Michael Strube. 2004. Semantic similarity applied to spoken dialogue summarization. In *Proceedings of Coling 2004*, pages 764–770, Geneva, Switzerland, August. COLING.
- Michael A.K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.
- Marti Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.
- Derrick Higgins, Jill Burstein, Daniel Marcu, and Claudia Gentile. 2004. Evaluating multiple aspects of coherence in student essays. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 185–192, Boston, Massachusetts, USA, May. Association for Computational Linguistics.
- Graeme Hirst and Alexander Budanitsky. 2005. Correcting real-word spelling errors by restoring lexical cohesion. *Natural Language Engineering*, 11(1):87–111.
- Michael Hoey. 1991. *Patterns of Lexis in Text*. Oxford University Press.
- Michael Hoey. 2005. *Lexical Priming*. Routledge.
- Diana Inkpen and Alain Désilets. 2005. Semantic similarity for detecting recognition errors in automatic speech transcripts. In *Proceedings of Empirical Methods in Natural Language Processing Conference*, pages 49–56, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.
- Michael Jones and Douglas Mewhort. 2007. Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114(1):1–37.
- Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of ACL*, pages 768–774, Montreal, Canada.
- Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments & Computers*, 28:203–208.
- Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA.
- Meghana Marathe and Graeme Hirst. 2010. Lexical Chains Using Distributional Measures of Concept Distance. In *Proceedings of 11th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING)*, pages 291–302, Iasi, Romania, March.
- Quinn McNemar. 1955. *Psychological Statistics*. New York: J. Wiley and Sons, 2nd edition.
- Eleni Miltsakaki and Karen Kukich. 2004. Evaluation of text coherence for electronic essay scoring systems. *Natural Language Engineering*, 10(1):25–55.
- Hemant Misra, François Yvon, Joemon M. Jose, and Olivier Cappe. 2009. Text segmentation via topic modeling: an analytical study. In *Proceedings of the 18th ACM conference on Information and knowledge management, CIKM ’09*, pages 1553–1556, New York, NY, USA. ACM.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 236–244, Columbus, Ohio, June. Association for Computational Linguistics.
- Jane Morris and Graeme Hirst. 1991. Lexical cohesion, the thesaurus, and the structure of text. *Computational linguistics*, 17(1):21–48.

- Sebastian Pado and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.
- Pavel Pecina. 2010. Lexical association measures and collocation extraction. *Language Resources and Evaluation*, 44:137–158.
- David Reshef, Yakir Reshef, Hilary Finucane, Sharon Grossman, Gilean McVean, Peter Turnbaugh, Eric Lander, Michael Mitzenmacher, and Pardis Sabeti. 2011. Detecting novel associations in large data sets. *Science*, 334(6062):1518–1524.
- Martin Riedl and Chris Biemann. 2012. How text segmentation algorithms gain from topic models. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 553–557, Montréal, Canada, June. Association for Computational Linguistics.
- Gerard Salton, Andrew Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.
- Kathy Sheehan, Irene Kostin, and Yoko Futagi. 2008. When do standard approaches for measuring vocabulary difficulty, syntactic complexity and referential cohesion yield biased estimates of text difficulty? In *Proceedings of the Cognitive Science Society*, pages 1978–1983, Washington, DC, July.
- Gregory Silber and Kathleen McCoy. 2002. Efficiently computed lexical chains as an intermediate representation for automatic text summarization. *Computational Linguistics*, 28(4):487–496.
- Mark Steyvers and Joshua B. Tenenbaum. 2005. The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. *Cognitive Science*, 29:41–78.
- Nicola Stokes, Joe Carthy, and Alan F. Smeaton. 2004. Select: A lexical cohesion based news story segmentation system. *Journal of AI Communications*, 17(1):3–12.
- Peter Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Peter D. Turney. 2001. Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In *European Conference on Machine Learning*, pages 491–502, Freiburg, Germany, September.
- Ziqi Zhang, Anna Gentile, and Fabio Ciravegna. 2012. Recent advances in methods of lexical semantic relatedness – a survey. *Natural Language Engineering*, FirstView:1–69.

# Adaptive Parser-Centric Text Normalization

Congle Zhang\*

Dept of Computer Science and Engineering  
University of Washington, Seattle, WA 98195, USA  
clzhang@cs.washington.edu

Tyler Baldwin   Howard Ho   Benny Kimelfeld   Yunyao Li

IBM Research - Almaden  
650 Harry Road, San Jose, CA 95120, USA  
{tbaldwi, ctho, kimelfeld, yunyaoli}@us.ibm.com

## Abstract

Text normalization is an important first step towards enabling many Natural Language Processing (NLP) tasks over informal text. While many of these tasks, such as parsing, perform the best over fully grammatically correct text, most existing text normalization approaches narrowly define the task in the *word-to-word* sense; that is, the task is seen as that of mapping all out-of-vocabulary non-standard words to their in-vocabulary standard forms. In this paper, we take a parser-centric view of normalization that aims to convert raw informal text into grammatically correct text. To understand the real effect of normalization on the parser, we tie normalization performance directly to parser performance. Additionally, we design a customizable framework to address the often overlooked concept of domain adaptability, and illustrate that the system allows for transfer to new domains with a minimal amount of data and effort. Our experimental study over datasets from three domains demonstrates that our approach outperforms not only the state-of-the-art word-to-word normalization techniques, but also manual word-to-word annotations.

## 1 Introduction

Text normalization is the task of transforming informal writing into its standard form in the language. It is an important processing step for a wide range of Natural Language Processing (NLP) tasks such as text-to-speech synthesis, speech recognition, information extraction, parsing, and machine translation (Sproat et al., 2001).

The use of normalization in these applications poses multiple challenges. First, as it is most often conceptualized, normalization is seen as the task of mapping all out-of-vocabulary non-standard word tokens to their in-vocabulary standard forms. However, the scope of the task can also be seen as much wider, encompassing whatever actions are required to convert the raw text into a fully grammatical sentence. This broader definition of the normalization task may include modifying punctuation and capitalization, and adding, removing, or reordering words. Second, as with other NLP techniques, normalization approaches are often focused on one primary domain of interest (e.g., Twitter data). Because the style of informal writing may be different in different data sources, tailoring an approach towards a particular data source can improve performance in the desired domain. However, this is often done at the cost of adaptability.

This work introduces a customizable normalization approach designed with domain transfer in mind. In short, customization is done by providing the normalizer with *replacement generators*, which we define in Section 3. We show that the introduction of a small set of domain-specific generators and training data allows our model to outperform a set of competitive baselines, including state-of-the-art word-to-word normalization. Additionally, the flexibility of the model also allows it to attempt to produce fully grammatical sentences, something not typically handled by word-to-word normalization approaches.

Another potential problem with state-of-the-art normalization is the lack of appropriate evaluation metrics. The normalization task is most frequently motivated by pointing to the need for clean text for downstream processing applications, such as syntactic parsing. However, most studies of normalization give little insight into whether and to what degree the normalization process improves

---

\*This work was conducted at IBM.

the performance of the downstream application. For instance, it is unclear how performance measured by the typical normalization evaluation metrics of word error rate and BLEU score (Papineni et al., 2002) translates into performance on a parsing task, where a well placed punctuation mark may provide more substantial improvements than changing a non-standard word form. To address this problem, this work introduces an evaluation metric that ties normalization performance directly to the performance of a downstream dependency parser.

The rest of this paper is organized as follows. In Section 2 we discuss previous approaches to the normalization problem. Section 3 presents our normalization framework, including the actual normalization and learning procedures. Our instantiation of this model is presented in Section 4. In Section 5 we introduce the parser driven evaluation metric, and present experimental results of our model with respect to several baselines in three different domains. Finally, we discuss our experimental study in Section 6 and conclude in Section 7.

## 2 Related Work

Sproat et al. (2001) took the first major look at the normalization problem, citing the need for normalized text for downstream applications. Unlike later works that would primarily focus on specific noisy data sets, their work is notable for attempting to develop normalization as a general process that could be applied to different domains. The recent rise of heavily informal writing styles such as Twitter and SMS messages set off a new round of interest in the normalization problem.

Research on SMS and Twitter normalization has been roughly categorized as drawing inspiration from three other areas of NLP (Kobus et al., 2008): machine translation, spell checking, and automatic speech recognition. The statistical machine translation (SMT) metaphor was the first proposed to handle the text normalization problem (Aw et al., 2006). In this mindset, normalizing SMS can be seen as a translation task from a source language (informal) to a target language (formal), which can be undertaken with typical noisy channel based models. Work by Choudhury et al. (2007) adopted the spell checking metaphor, casting the problem in terms of character-level, rather than word-level, edits. They proposed an HMM based model that

takes into account both grapheme and phoneme information. Kobus et al. (2008) undertook a hybrid approach that pulls inspiration from both the machine translation and speech recognition metaphors.

Many other approaches have been examined, most of which are at least partially reliant on the above three metaphors. Cook and Stevenson (2009) perform an unsupervised method, again based on the noisy channel model. Pennell and Liu (2011) developed a CRF tagger for deletion-based abbreviation on tweets. Xue et al. (2011) incorporated orthographic, phonetic, contextual, and acronym expansion factors to normalize words in both Twitter and SMS. Liu et al. (2011) modeled the generation process from dictionary words to non-standard tokens under an unsupervised sequence labeling framework. Han and Baldwin (2011) use a classifier to detect ill-formed words, and then generate correction candidates based on morphophonemic similarity. Recent work has looked at the construction of normalization dictionaries (Han et al., 2012) and on improving coverage by integrating different human perspectives (Liu et al., 2012).

Although it is almost universally used as a motivating factor, most normalization work does not directly focus on improving downstream applications. While a few notable exceptions highlight the need for normalization as part of text-to-speech systems (Beaufort et al., 2010; Pennell and Liu, 2010), these works do not give any direct insight into how much the normalization process actually improves the performance of these systems. To our knowledge, the work presented here is the first to clearly link the output of a normalization system to the output of the downstream application. Similarly, our work is the first to prioritize domain adaptation during the new wave of text message normalization.

## 3 Model

In this section we introduce our normalization framework, which draws inspiration from our previous work on spelling correction for search (Bao et al., 2011).

### 3.1 Replacement Generators

Our input the original, unnormalized text, represented as a sequence  $\mathbf{x} = x_1, x_2, \dots, x_n$  of *tokens*  $x_i$ . In this section we will use the following se-

quence as our running example:

$\mathbf{x} = \text{Ay}_1 \text{ wou}d\text{ent}_2 \text{ of}_3 \text{ see}_4 \text{ 'em}_5$

where space replaces comma for readability, and each token is subscripted by its position. Given the input  $\mathbf{x}$ , we apply a series of replacement generators, where a *replacement generator* is a function that takes  $\mathbf{x}$  as input and produces a collection of replacements. Here, a *replacement* is a statement of the form “replace tokens  $x_i, \dots, x_{j-1}$  with  $\mathbf{s}$ .” More precisely, a replacement is a triple  $\langle i, j, \mathbf{s} \rangle$ , where  $1 \leq i \leq j \leq n + 1$  and  $\mathbf{s}$  is a sequence of tokens. Note that in the case where  $i = j$ , the sequence  $\mathbf{s}$  should be *inserted* right before  $x_i$ ; and in the special case where  $\mathbf{s}$  is empty, we simply delete  $x_i, \dots, x_{j-1}$ . For instance, in our running example the replacement  $\langle 2, 3, \text{would not} \rangle$  replaces  $x_2 = \text{wou}d\text{ent}$  with  $\text{would not}$ ;  $\langle 1, 2, \text{Ay} \rangle$  replaces  $x_1$  with itself (hence, does not change  $\mathbf{x}$ );  $\langle 1, 2, \epsilon \rangle$  (where  $\epsilon$  is the empty sequence) deletes  $x_1$ ;  $\langle 6, 6, . \rangle$  inserts a period at the end of the sequence.

The provided replacement generators can be either generic (cross domain) or domain-specific, allowing for domain customization. In Section 4, we discuss the replacement generators used in our empirical study.

### 3.2 Normalization Graph

Given the input  $\mathbf{x}$  and the set of replacements produced by our generators, we associate a unique Boolean variable  $X_r$  with each replacement  $r$ . As expected,  $X_r$  being **true** means that the replacement  $r$  takes place in producing the output sequence.

Next, we introduce dependencies among variables. We first discuss the syntactic consistency of truth assignments. Let  $r_1 = \langle i_1, j_1, \mathbf{s}_1 \rangle$  and  $r_2 = \langle i_2, j_2, \mathbf{s}_2 \rangle$  be two replacements. We say that  $r_1$  and  $r_2$  are *locally consistent* if the intervals  $[i_1, j_1)$  and  $[i_2, j_2)$  are disjoint. Moreover, we do not allow two insertions to take place at the same position; therefore, we exclude  $[i_1, j_1)$  and  $[i_2, j_2)$  from the definition of local consistency when  $i_1 = j_1 = i_2 = j_2$ . If  $r_1$  and  $r_2$  are locally consistent and  $j_1 = i_2$ , then we say that  $r_2$  is a *consistent follower* of  $r_1$ .

A truth assignment  $\alpha$  to our variables  $X_r$  is *sound* if every two replacements  $r$  and  $r'$  with  $\alpha(X_r) = \alpha(X_{r'}) = \mathbf{true}$  are locally consistent. We say that  $\alpha$  is *complete* if every token

of  $\mathbf{x}$  is captured by at least one replacement  $r$  with  $\alpha(X_r) = \mathbf{true}$ . Finally, we say that  $\alpha$  is *legal* if it is sound and complete. The output (normalized sequence) defined by a legal assignment  $\alpha$  is, naturally, the concatenation (from left to right) of the strings  $\mathbf{s}$  in the replacements  $r = \langle i, j, \mathbf{s} \rangle$  with  $\alpha(X_r) = \mathbf{true}$ . In Figure 1, for example, if the nodes with a grey shade are the ones associated with **true** variables under  $\alpha$ , then the output defined by  $\alpha$  is I would not have seen them.

Our variables carry two types of interdependencies. The first is that of syntactic consistency: the entire assignment is required to be legal. The second captures correlation among replacements. For instance, if we replace `of` with `have` in our running example, then the next `see` token is more likely to be replaced with `seen`. In this work, dependencies of the second type are restricted to pairs of variables, where each pair corresponds to a replacement and a consistent follower thereof.

The above dependencies can be modeled over a standard undirected graph using Conditional Random Fields (Lafferty et al., 2001). However, the graph would be complex: in order to model local consistency, there should be edges between every two nodes that violate local consistency. Such a model renders inference and learning infeasible. Therefore, we propose a clearer model by a directed graph, as illustrated in Figure 1 (where nodes are represented by replacements  $r$  instead of the variables  $X_r$ , for readability). To incorporate correlation among replacements, we introduce an edge from  $X_r$  to  $X_{r'}$  whenever  $r'$  is a consistent follower of  $r$ . Moreover, we introduce two dummy nodes, `start` and `end`, with an edge from `start` to each variable that corresponds to a prefix of the input sequence  $\mathbf{x}$ , and an edge from each variable that corresponds to a suffix of  $\mathbf{x}$  to `end`.

The principal advantage of modeling the dependencies in such a directed graph is that now, the legal assignments are in one-to-one correspondence with the paths from `start` to `end`; this is a straightforward observation that we do not prove here.

We appeal to the log-linear model formulation to define the probability of an assignment. The conditional probability of an assignment  $\alpha$ , given an input sequence  $\mathbf{x}$  and the weight vector  $\Theta = \langle \theta_1, \dots, \theta_k \rangle$  for our features, is defined as  $p(\alpha |$

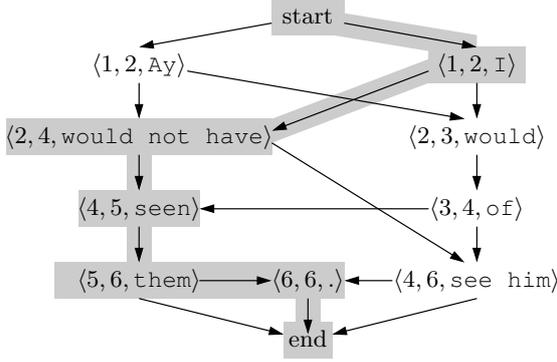


Figure 1: Example of a normalization graph; the nodes are replacements generated by the replacement generators, and every path from start to end implies a legal assignment

$\mathbf{x}, \Theta) = 0$  if  $\alpha$  is not legal, and otherwise,

$$p(\alpha \mid \mathbf{x}, \Theta) = \frac{1}{Z(\mathbf{x})} \prod_{X \rightarrow Y \in \alpha} \exp\left(\sum_j \theta_j \phi_j(X, Y, \mathbf{x})\right).$$

Here,  $Z(\mathbf{x})$  is the partition function,  $X \rightarrow Y \in \alpha$  refers to an edge  $X \rightarrow Y$  with  $\alpha(X) = \mathbf{true}$  and  $\alpha(Y) = \mathbf{true}$ , and  $\phi_1(X, Y, \mathbf{x}), \dots, \phi_k(X, Y, \mathbf{x})$  are real valued feature functions that are weighted by  $\theta_1, \dots, \theta_k$  (the model’s parameters), respectively.

### 3.3 Inference

When performing inference, we wish to select the output sequence with the highest probability, given the input sequence  $\mathbf{x}$  and the weight vector  $\Theta$  (i.e., MAP inference). Specifically, we want an assignment  $\alpha^* = \arg \max_{\alpha} p(\alpha \mid \mathbf{x}, \Theta)$ .

While exact inference is computationally hard on general graph models, in our model it boils down to finding the longest path in a weighted and acyclic directed graph. Indeed, our directed graph (illustrated in Figure 1) is acyclic. We assign the real value  $\sum_j \theta_j \phi_j(X, Y, \mathbf{x})$  to the edge  $X \rightarrow Y$ , as the weight. As stated in Section 3.2, a legal assignment  $\alpha$  corresponds to a path from start to end; moreover, the sum of the weights on that path is equal to  $\log p(\alpha \mid \mathbf{x}, \Theta) + \log Z(\mathbf{x})$ . In particular, a longer path corresponds to an assignment with greater probability. Therefore, we can solve the MAP inference within our model by finding the weighted longest path in the directed acyclic graph. The algorithm in Figure 2 summarizes the inference procedure to normalize the input sequence  $\mathbf{x}$ .

#### Input:

1. A sequence  $\mathbf{x}$  to normalize;
2. A weight vector  $\Theta = \langle \theta_1, \dots, \theta_k \rangle$ .

**Generate replacements:** Apply all replacement generators to get a set of replacements  $r$ , each  $r$  is a triple  $\langle i, j, s \rangle$ .

#### Build a normalization graph:

1. For each replacement  $r$ , create a node  $X_r$ .
2. For each  $r'$  and  $r$ , create an edge  $X_r$  to  $X_{r'}$  if  $r'$  is a consistent follower of  $r$ .
3. Create two dummy nodes start and end, and create edges from start to all prefix nodes and end to all suffix nodes.
4. For each edge  $X \rightarrow Y$ , compute the features  $\phi_j(X, Y, \mathbf{x})$ , and weight the edge by  $\sum_j \theta_j \phi_j(X, Y, \mathbf{x})$ .

**MAP Inference:** Find a weighted longest path  $P$  from start to end, and return  $\alpha^*$ , where  $\alpha^*(X_r) = \mathbf{true}$  iff  $X_r \in P$ .

Figure 2: Normalization algorithm

### 3.4 Learning

Our labeled data consists of pairs  $(\mathbf{x}_i, \mathbf{y}_i^{\text{gold}})$ , where  $\mathbf{x}_i$  is an input sequence (to normalize) and  $\mathbf{y}_i^{\text{gold}}$  is a (manually) normalized sequence. We obtain a truth assignment  $\alpha_i^{\text{gold}}$  from each  $\mathbf{y}_i^{\text{gold}}$  by selecting an assignment  $\alpha$  that minimizes the edit distance between  $\mathbf{y}_i^{\text{gold}}$  and the normalized text implied by  $\alpha$ :

$$\alpha_i^{\text{gold}} = \arg \min_{\alpha} \text{DIST}(y(\alpha), \mathbf{y}_i^{\text{gold}}) \quad (1)$$

Here,  $y(\alpha)$  denotes the normalized text implied by  $\alpha$ , and DIST is a token-level edit distance. We apply a simple dynamic-programming algorithm to compute  $\alpha_i^{\text{gold}}$ . Finally, the items in our training data are the pairs  $(\mathbf{x}_i, \alpha_i^{\text{gold}})$ .

Learning over similar models is commonly done via maximum likelihood estimation:

$$L(\Theta) = \log \prod_i p(\alpha_i = \alpha_i^{\text{gold}} \mid \mathbf{x}_i, \Theta)$$

Taking the partial derivative gives the following:

$$\sum_i \left( \Phi_j(\alpha_i^{\text{gold}}, \mathbf{x}_i) - E_{p(\alpha_i \mid \mathbf{x}_i, \Theta)} \Phi_j(\alpha_i, \mathbf{x}_i) \right)$$

where  $\Phi_j(\alpha, \mathbf{x}) = \sum_{X \rightarrow Y} \phi_j(X, Y, \mathbf{x})$ , that is, the sum of values for the  $j$ th feature along the

<p><b>Input:</b></p> <ol style="list-style-type: none"> <li>1. A set <math>\{(\mathbf{x}_i, \mathbf{y}_i^{\text{gold}})\}_{i=1}^n</math> of sequences and their gold normalization;</li> <li>2. Number <math>T</math> of iterations.</li> </ol> <p><b>Initialization:</b> Initialize each <math>\theta_j</math> as zero, and obtain each <math>\alpha_i^{\text{gold}}</math> according to (1).</p> <p><b>Repeat <math>T</math> times:</b></p> <ol style="list-style-type: none"> <li>1. Infer each <math>\alpha_i^*</math> from <math>\mathbf{x}_i</math> using the current <math>\Theta</math>;</li> <li>2. <math>\theta_j \leftarrow \theta_j + \sum_i (\Phi_j(\alpha_i^{\text{gold}}, \mathbf{x}_i) - \Phi_j(\alpha_i^*, \mathbf{x}_i))</math> for all <math>j = 1, \dots, k</math>.</li> </ol> <p><b>Output:</b> <math>\Theta = \langle \theta_1, \dots, \theta_k \rangle</math></p>
--

Figure 3: Learning algorithm

path defined by  $\alpha$ , and  $E_{p(\alpha_i|\mathbf{x}_i, \Theta)} \Phi_j(\alpha_i, \mathbf{x}_i)$  is the expected value of that sum (over all legal assignments  $\alpha_i$ ), assuming the current weight vector.

How to efficiently compute  $E_{p(\alpha_i|\mathbf{x}_i, \Theta)} \Phi_j(\alpha_i, \mathbf{x}_i)$  in our model is unclear; naively, it requires enumerating all legal assignments. We instead opt to use a more tractable perceptron-style algorithm (Collins, 2002). Instead of computing the expectation, we simply compute  $\Phi_j(\alpha_i^*, \mathbf{x}_i)$ , where  $\alpha_i^*$  is the assignment with the highest probability, generated using the current weight vector. The result is then:

$$\sum_i \left( \Phi_j(\alpha_i^{\text{gold}}, \mathbf{x}_i) - \Phi_j(\alpha_i^*, \mathbf{x}_i) \right)$$

Our learning applies the following two steps iteratively. (1) Generate the most probable sequence within the current weights. (2) Update the weights by comparing the path generated in the previous step to the gold standard path. The algorithm in Figure 3 summarizes the procedure.

## 4 Instantiation

In this section, we discuss our instantiation of the model presented in the previous section. In particular, we describe our replacement generators and features.

### 4.1 Replacement Generators

One advantage of our proposed model is that the reliance on replacement generators allows for strong flexibility. Each generator can be seen as a black box, allowing replacements that are created heuristically, statistically, or by external tools to be incorporated within the same framework.

Generator	From	To
leave intact	good	good
edit distance	bac	back
lowercase	NEED	need
capitalize	it	It
Google spell	disspaear	disappear
contraction	wouldn't	would not
slang language	ima	I am going to
insert punctuation	€	.
duplicated punctuation	!?	!
delete filler	lmao	€

Table 1: Example replacement generators

To build a set of generic replacement generators suitable for normalizing a variety of data types, we collected a set of about 400 Twitter posts as development data. Using that data, a series of generators were created; a sample of them are shown in Table 1. As shown in the table, these generators cover a variety of normalization behavior, from changing non-standard word forms to inserting and deleting tokens.

### 4.2 Features

Although the proposed framework supports real valued features, all features in our system are binary. In total, we used 70 features. Our feature set pulls information from several different sources:

**N-gram:** Our n-gram features indicate the frequency of the phrases induced by an edge. These features are turned into binary ones by bucketing their log values. For example, on the edge from  $\langle 1, 2, \text{I} \rangle$  to  $\langle 2, 3, \text{would} \rangle$  such a feature will indicate whether the frequency of `I would` is over a threshold. We use the Corpus of Contemporary English (Davies, 2008) to produce our n-gram information.

**Part-of-speech:** Part-of-speech information can be used to produce features that encourage certain behavior, such as avoiding the deletion of noun phrases. We generate part-of-speech information over the original raw text using a Twitter part-of-speech tagger (Ritter et al., 2011). Of course, the part-of-speech information obtained this way is likely to be noisy, and we expect our learning algorithm to take that into account.

**Positional:** Information from positions is used primarily to handle capitalization and punctuation insertion, for example, by incorporating features for capitalized words after stop punctuation or the insertion of stop punctuation at the end of the sentence.

**Lineage:** Finally, we include binary features

that indicate which generator spawned the replacement.

## 5 Evaluation

In this section, we present an empirical study of our framework. The study is done over datasets from three different domains. The goal is to evaluate the framework in two aspects: (1) usefulness for downstream applications (specifically dependency parsing), and (2) domain adaptability.

### 5.1 Evaluation Metrics

A few different metrics have been used to evaluate normalizer performance, including word error rate and BLEU score. While each metric has its pros and cons, they all rely on word-to-word matching and treat each word equally. In this work, we aim to evaluate the performance of a normalizer based on how it affects the performance of downstream applications. We find that the conventional metrics are not directly applicable, for several reasons. To begin with, the assumption that words have equal weights is unlikely to hold. Additionally, these metrics tend to ignore other important non-word information such as punctuation or capitalization. They also cannot take into account other aspects that may have an impact on downstream performance, such as the word reordering as seen in the example in Figure 4. Therefore, we propose a new evaluation metric that directly equates normalization performance with the performance of a common downstream application—dependency parsing.

To realize our desired metric, we apply the following procedure. First, we produce gold standard normalized data by manually normalizing sentences to their full grammatically correct form. In addition to the word-to-word mapping performed in typical normalization gold standard generation, this annotation procedure includes all actions necessary to make the sentence grammatical, such as word reordering, modifying capitalization, and removing emoticons. We then run an off-the-shelf dependency parser on the gold standard normalized data to produce our gold standard parses. Although the parser could still produce mistakes on the grammatical sentences, we feel that this provides a realistic benchmark for comparison, as it represents an upper bound on the possible performance of the parser, and avoids an expensive second round of manual annotation.

Test	Gold	SVO
<i>I kinda wanna get ipad NEW</i>	<i>I kind of want to get a new iPad.</i>	
<b>verb(get)</b>	verb(want) <b>verb(get)</b>	$\text{precision}_v = \frac{1}{1}$ $\text{recall}_v = \frac{1}{2}$
<b>subj(get,I)</b> subj(get,wanna) obj(get,NEW)	subj(want,I) <b>subj(get,I)</b> obj(get,iPad)	$\text{precision}_{so} = \frac{1}{3}$ $\text{recall}_{so} = \frac{1}{3}$

Figure 4: The subjects, verbs, and objects identified on example test/gold text, and corresponding metric scores

To compare the parses produced over automatically normalized data to the gold standard, we look at the subjects, verbs, and objects (SVO) identified in each parse. The metric shown in Equations (2) and (3) below is based on the identified subjects and objects in those parses. Note that  $SO$  denotes the set of identified subjects and objects whereas  $SO^{\text{gold}}$  denotes the set of subjects and objects identified when parsing the gold-standard normalization.

$$\text{precision}_{so} = \frac{|SO \cap SO^{\text{gold}}|}{|SO|} \quad (2)$$

$$\text{recall}_{so} = \frac{|SO \cap SO^{\text{gold}}|}{|SO^{\text{gold}}|} \quad (3)$$

We similarly define  $\text{precision}_v$  and  $\text{recall}_v$ , where we compare the set  $V$  of identified verbs to  $V^{\text{gold}}$  of those found in the gold-standard normalization. An example is shown in Figure 4.

### 5.2 Results

To establish the extensibility of our normalization system, we present results in three different domains: Twitter posts, Short Message Service (SMS) messages, and call-center logs. For Twitter and SMS messages, we used established datasets to compare with previous work. As no established call-center log dataset exists, we collected our own. In each case, we ran the proposed system with two different configurations: one using only the generic replacement generators presented in Section 4 (denoted as *generic*), and one that adds additional domain-specific generators for the corresponding domain (denoted as *domain-specific*). All runs use ten-fold cross validation for training and evaluation. The Stanford parser<sup>1</sup> (Marneffe et al., 2006) was used to produce all dependency

<sup>1</sup>Version 2.0.4, <http://nlp.stanford.edu/software/lex-parser.shtml>

parses. We compare our system to the following baseline solutions:

**w/oN:** No normalization is performed.

**Google:** Output of the Google spell checker.

**w2wN:** The output of the word-to-word normalization of Han and Baldwin (2011). Not available for call-center data.

**Gw2wN:** The manual gold standard word-to-word normalizations of previous work (Choudhury et al., 2007; Han and Baldwin, 2011). Not available for call-center data.

Our results use the metrics of Section 5.1.

### 5.2.1 Twitter

To evaluate the performance on Twitter data, we use the dataset of randomly sampled tweets produced by (Han and Baldwin, 2011). Because the gold standard used in this work only provided word mappings for out-of-vocabulary words and did not enforce grammaticality, we reannotated the gold standard data<sup>2</sup>. Their original gold standard annotations were kept as a baseline.

To produce Twitter-specific generators, we examined the Twitter development data collected for generic generator production (Section 4). These generators focused on the Twitter-specific notions of hashtags (#), ats (@), and retweets (RT). For each case, we implemented generators that allowed for either the initial symbol or the entire token to be deleted (e.g., @Hertz to Hertz, @Hertz to  $\epsilon$ ).

The results are given in Table 2. As shown, the domain-specific generators yielded performance significantly above the generic ones and all baselines. Even without domain-specific generators, our system outperformed the word-to-word normalization approaches. Most notably, both the generic and domain-specific systems outperformed the gold standard word-to-word normalizations. These results validate the hypothesis that simple word-to-word normalization is insufficient if the goal of normalization is to improve dependency parsing; even if a system could produce perfect word-to-word normalization, it would produce lower quality parses than those produced by our approach.

<sup>2</sup>Our results and the reannotations of the Twitter and SMS data are available at <https://www.cs.washington.edu/node/9091/>

System	Verb			Subject-Object		
	Pre	Rec	F1	Pre	Rec	F1
w/oN	83.7	68.1	75.1	31.7	38.6	34.8
Google	88.9	78.8	83.5	36.1	46.3	40.6
w2wN	87.5	81.5	84.4	44.5	58.9	50.7
Gw2w	89.8	83.8	86.7	46.9	61.0	53.0
generic	91.7	88.9	90.3	53.6	70.2	60.8
domain specific	95.3	88.7	<b>91.9</b>	72.5	76.3	<b>74.4</b>

Table 2: Performance on Twitter dataset

### 5.2.2 SMS

To evaluate the performance on SMS data, we use the Treasure My Text data collected by Choudhury et al. (2007). As with the Twitter data, the word-to-word normalizations were reannotated to enforce grammaticality. As a replacement generator for SMS-specific substitutions, we used a mapping dictionary of SMS abbreviations.<sup>3</sup> No further SMS-specific development data was needed.

Table 3 gives the results on the SMS data. The SMS dataset proved to be more difficult than the Twitter dataset, with the overall performance of every system being lower. While this drop of performance may be a reflection of the difference in data styles between SMS and Twitter, it is also likely a product of the collection methodology. The collection methodology of the Treasure My Text dataset dictated that every message must have at least one mistake, which may have resulted in a dataset that was noisier than average.

Nonetheless, the trends on SMS data mirror those on Twitter data, with the domain-specific generators achieving the greatest overall performance. However, while the generic setting still manages to outperform most baselines, it did not outperform the gold word-to-word normalization. In fact, the gold word-to-word normalization was much more competitive on this data, outperforming even the domain-specific system on verbs alone. This should not be seen as surprising, as word-to-word normalization is most likely to be beneficial for cases like this where the proportion of non-standard tokens is high.

It should be noted that the SMS dataset as available has had all punctuation removed. While this may be appropriate for word-to-word normalization, this preprocessing may have an effect on the parse of the sentence. As our system has the ability to add punctuation but our baseline systems do not, this has the potential to artificially inflate our results. To ensure a fair comparison, we manually

<sup>3</sup><http://www.netlingo.com/acronyms.php>

System	Verb			Subject-Object		
	Rec	Pre	F1	Rec	Pre	F1
w/oN	76.4	48.1	59.0	19.5	21.5	20.4
Google	85.1	61.6	71.5	22.4	26.2	24.1
w2wN	78.5	61.5	68.9	29.9	36.0	32.6
Gw2wN	87.6	76.6	<b>81.8</b>	38.0	50.6	43.4
generic	86.5	77.4	81.7	35.5	47.7	40.7
domain specific	88.1	75.0	81.0	41.0	49.5	<b>44.8</b>

Table 3: Performance on SMS dataset

System	Verb			Subject-Object		
	Pre	Rec	F1	Pre	Rec	F1
w/oN	98.5	97.1	97.8	69.2	66.1	67.6
Google	99.2	97.9	<b>98.5</b>	70.5	67.3	68.8
generic	98.9	97.4	98.1	71.3	67.9	69.6
domain specific	99.2	97.4	98.3	87.9	83.1	<b>85.4</b>

Table 4: Performance on call-center dataset

added punctuation to a randomly selected small subset of the SMS data and reran each system. This experiment suggested that, in contrast to the hypothesis, adding punctuation actually improved the results of the proposed system more substantially than that of the baseline systems.

### 5.2.3 Call-Center

Although Twitter and SMS data are unmistakably different, there are many similarities between the two, such as the frequent use of shorthand word forms that omit letters. The examination of call-center logs allows us to examine the ability of our system to perform normalization in more disparate domains. Our call-center data consists of text-based responses to questions about a user’s experience with a call-center (e.g., their overall satisfaction with the service). We use call-center logs from a major company, and collect about 150 responses for use in our evaluation. We collected an additional small set of data to develop our call-center-specific generators.

Results on the call-center dataset are in Table 4. As shown, the raw call-center data was comparatively clean, resulting in higher baseline performance than in other domains. Unlike on previous datasets, the use of generic mappings only provided a small improvement over the baseline. However, the use of domain-specific generators once again led to significantly increased performance on subjects and objects.

## 6 Discussion

The results presented in the previous section suggest that domain transfer using the proposed nor-

malization framework is possible with only a small amount of effort. The relatively modest set of additional replacement generators included in each data set allowed the domain-specific approaches to significantly outperform the generic approach. In the call-center case, performance improvements could be seen by referencing a very small amount of development data. In the SMS case, the presence of a domain-specific dictionary allowed for performance improvements without the need for any development data at all. It is likely, though not established, that employing further development data would result in further performance improvements. We leave further investigation to future work.

The results in Section 5.2 establish a point that has often been assumed but, to the best of our knowledge, has never been explicitly shown: performing normalization is indeed beneficial to dependency parsing on informal text. The parse of the normalized text was substantially better than the parse of the original raw text in all domains, with absolute performance increases ranging from about 18-25% on subjects and objects. Furthermore, the results suggest that, as hypothesized, preparing an informal text for a parsing task requires more than simple word-to-word normalization. The proposed approach significantly outperforms the state-of-the-art word-to-word normalization approach. Perhaps most interestingly, the proposed approach performs on par with, and in several cases superior to, gold standard word-to-word annotations. This result gives strong evidence for the conclusion that parser-targeted normalization requires a broader understanding of the scope of the normalization task.

While the work presented here gives promising results, there are still many behaviors found in informal text that prove challenging. One such example is the word reordering seen in Figure 4. Although word reordering could be incorporated into the model as a combination of a deletion and an insertion, the model as currently devised cannot easily link these two replacements to one another. Additionally, instances of reordering proved hard to detect in practice. As such, no reordering-based replacement generators were implemented in the presented system. Another case that proved difficult was the insertion of missing tokens. For instance, the informal sentence “Day 3 still don’t freaking

feel good!:(” could be formally rendered as “**It is** day 3 **and I** still do not feel good!”. Attempts to address missing tokens in the model resulted in frequent false positives. Similarly, punctuation insertion proved to be challenging, often requiring a deep analysis of the sentence. For example, contrast the sentence “I’m watching a movie I don’t know its name.” which would benefit from inserted punctuation, with “I’m watching a movie I don’t know.”, which would not. We feel that the work presented here provides a foundation for future work to more closely examine these challenges.

## 7 Conclusions

This work presents a framework for normalization with an eye towards domain adaptation. The proposed framework builds a statistical model over a series of replacement generators. By doing so, it allows a designer to quickly adapt a generic model to a new domain with the inclusion of a small set of domain-specific generators. Tests over three different domains suggest that, using this model, only a small amount of domain-specific data is necessary to tailor an approach towards a new domain.

Additionally, this work introduces a parser-centric view of normalization, in which the performance of the normalizer is directly tied to the performance of a downstream dependency parser. This evaluation metric allows for a deeper understanding of how certain normalization actions impact the output of the parser. Using this metric, this work established that, when dependency parsing is the goal, typical word-to-word normalization approaches are insufficient. By taking a broader look at the normalization task, the approach presented here is able to outperform not only state-of-the-art word-to-word normalization approaches but also manual word-to-word annotations.

Although the work presented here established that more than word-to-word normalization was necessary to produce parser-ready normalizations, it remains unclear which specific normalization tasks are most critical to parser performance. We leave this interesting area of examination to future work.

## Acknowledgments

We thank the anonymous reviewers of ACL for helpful comments and suggestions. We also thank Ioana R. Stanoi for her comments on a preliminary version of this work, Daniel S. Weld for his support, and Alan Ritter, Monojit Choudhury, Bo Han, and Fei Liu for sharing their tools and data. The first author is partially supported by the DARPA Machine Reading Program under AFRL prime contract numbers FA8750-09-C-0181 and FA8750-09-C-0179. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, AFRL, or the US government. This work is a part of IBM’s SystemT project (Chiticariu et al., 2010).

## References

- AiTī Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for sms text normalization. In *ACL*, pages 33–40.
- Zhuowei Bao, Benny Kimelfeld, and Yunyao Li. 2011. A graph approach to spelling correction in domain-centric search. In *ACL*, pages 905–914.
- Richard Beaufort, Sophie Roekhaut, Louise-Amélie Cougnon, and Cédric Fairon. 2010. A hybrid rule/model-based finite-state framework for normalizing sms messages. In *ACL*, pages 770–779.
- Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. SystemT: An algebraic approach to declarative information extraction. In *ACL*, pages 128–137.
- Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *IJDAR*, 10(3-4):157–174.
- Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, pages 1–8.
- Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text message normalization. In *CALC*, pages 71–78.
- Mark Davies. 2008-. The corpus of contemporary american english: 450 million words, 1990-present. Available online at: <http://corpus.byu.edu/coca/>.
- Bo Han and Timothy Baldwin. 2011. Lexical normalization of short text messages: Makn sens a #twitter. In *ACL*, pages 368–378.

- Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *EMNLP-CoNLL*, pages 421–432.
- Catherine Kobus, François Yvon, and Géraldine Damnati. 2008. Normalizing SMS: are two metaphors better than one? In *COLING*, pages 441–448.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.
- Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011. Insertion, deletion, or substitution? normalizing text messages without pre-categorization nor supervision. In *ACL*, pages 71–76.
- Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *ACL*, pages 1035–1044.
- Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC-06*, pages 449–454.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.
- Deana Pennell and Yang Liu. 2010. Normalization of text messages for text-to-speech. In *ICASSP*, pages 4842–4845.
- Deana Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of SMS abbreviations. In *IJCNLP*, pages 974–982.
- Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in Tweets: An experimental study. In *EMNLP*, pages 1524–1534.
- Richard Sproat, Alan W. Black, Stanley F. Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333.
- Zhenzhen Xue, Dawei Yin, and Brian D. Davison. 2011. Normalizing microtext. In *Analyzing Microtext*, volume WS-11-05 of *AAAI Workshops*.

# A Random Walk Approach to Selectional Preferences Based on Preference Ranking and Propagation\*

Zhenhua Tian<sup>†</sup>, Hengheng Xiang, Ziqi Liu, Qinghua Zheng<sup>‡</sup>

Ministry of Education Key Lab for Intelligent Networks and Network Security

Department of Computer Science and Technology

Xi'an Jiaotong University

Xi'an, Shaanxi 710049, China

{zhhtian<sup>†</sup>, qhzheng<sup>‡</sup>}@mail.xjtu.edu.cn

## Abstract

This paper presents an unsupervised random walk approach to alleviate data sparsity for selectional preferences. Based on the measure of preferences between predicates and arguments, the model aggregates all the transitions from a given predicate to its nearby predicates, and propagates their argument preferences as the given predicate's smoothed preferences. Experimental results show that this approach outperforms several state-of-the-art methods on the pseudo-disambiguation task, and it better correlates with human plausibility judgements.

## 1 Introduction

Selectional preferences (SP) or selectional restrictions capture the plausibility of predicates and their arguments for a given relation. Kaze and Fodor (1963) describe that predicates and their arguments have strict *boolean restrictions*, either satisfied or violated. Sentences are semantically anomalous and not consistent in reading if they violated the restrictions. Wilks (1973) argues that “rejecting utterances is just what humans do not. They try to understand them.” He further states selectional restrictions as *preferences* between the predicates and arguments, where the violation can be less preferred, but not fatal. For instance, given the predicate word *eat*, word *food* is likely to be its object, *iPhone* is likely to be implausible for it, and *tiger* is less preferred but not curious.

SP have been proven to help many natural language processing tasks that involve attachment de-

terminations, such as semantic role labeling (Resnik, 1993; Gildea and Jurafsky, 2002), word sense disambiguation (Resnik, 1997), human plausibility judgements (Spasić and Ananiadou, 2004), syntactic disambiguation (Toutanova et al., 2005), word compositionality (McCarthy et al., 2007), textual entailment (Pantel et al., 2007) and pronoun resolution (Bergsma et al., 2008) etc.

A direct approach to acquire SP is to extract triples  $(q, r, a)$  of predicates, relations, and arguments from a syntactically analyzed corpus, and then conduct maximum likelihood estimation (MLE) on the data. However, this strategy is infeasible for many plausible triples due to data sparsity. For example, given the relation  $\langle \textit{verb-dobj-noun} \rangle$  in a corpus, we may see plausible triples:

*eat* - {*food, cake, apple, banana, candy...*}

But we may not see plausible and implausible triples such as:

*eat* - {*watermelon, ziti, escarole, iPhone...*}

Then how to use a smooth model to alleviate data sparsity for SP?

Random walk models have been successfully applied to alleviate the data sparsity issue on collaborative filtering in recommender systems. Many online businesses, such as Netflix, Amazon.com, and Facebook, have used recommender systems to provide personalized suggestions on the movies, books, or friends that the users may prefer and interested in (Liben-Nowell and Kleinberg, 2007; Yildirim and Krishnamoorthy, 2008).

In this paper, we present an extension of using the random walk model to alleviate data sparsity for SP. The main intuition is to aggregate all the transitions from a given predicate to its nearby predicates, and propagate their preferences on arguments as the given predicate's smoothed argu-

\*Partial of this work was done when the first author visiting at Language Technologies Institute of Carnegie Mellon University sponsored by the China Scholarship Council.

ment preferences. Our work and contributions are summarized as follows:

- We present a framework of random walk approach to SP. It contains four components with *flexible* configurations. Each component is corresponding to a specific functional operation on the bipartite and monopartite graphs which representing the SP data;
- We propose an adjusted *preference ranking* method to measure SP based on the popularity and association of predicate-argument pairs. It better correlates with human plausibility judgements. It also helps to discover similar predicates more precisely;
- We introduce a *probability function* for random walk based on the predicate distances. It controls the influence of nearby and distant predicates to achieve more accurate results;
- We find out that *propagate* the measured preferences of predicate-argument pairs is more proper and natural for SP smooth. It helps to improve the final performance significantly.

We conduct experiments using two sections of the LDC English gigaword corpora as the generalization data. For the *pseudo-disambiguation* task, we evaluate it on the Penn TreeBank-3 data. Results show that our model outperforms several previous methods. We further investigate the correlations of smoothed scores with *human plausibility judgements*. Again our method achieves better correlations on two third party data.

The remainder of the paper is organized as follows: Section 2 introduces related work. Section 3 briefly formulates the overall framework of our method. Section 4 describes the detailed model configurations, with discussions on their roles and implications. Section 5 provides experiments on both the pseudo-disambiguation task and human plausibility judgements. Finally, Section 6 summarizes the conclusions and future work.

## 2 Related Work

### 2.1 WordNet-based Approach

Resnik (1996) conducts the pioneer work on corpus-driven SP induction. For a given predicate  $q$ , the system firstly computes its distribution of argument semantic classes based on WordNet. Then for a given argument  $a$ , the system collects

the set of candidate semantic classes which contain the argument  $a$ , and ensures they are seen in  $q$ . Finally the system picks a semantic class from the candidates with the maximal selectional association score, and defines the score as smoothed score of  $(q, a)$ .

Many researchers have followed the so-called WordNet-based approach to SP. One of the key issues is to induce the set of argument semantic classes that are acceptable by the given predicate. Li and Abe (1998) propose a tree cut model based on minimal description length (MDL) principle for the induction of semantic classes. Clark and Weir (2002) suggest a hypothesis testing method by ascending the noun hierarchy of WordNet. Ciaramita and Johnson (2000) model WordNet as a Bayesian network to solve the “explain away” ambiguity. Beyond induction on argument classes only, Agirre and Martinez (2001) propose a class-to-class model that simultaneously learns SP on both the predicate and argument classes.

WordNet-based approach produces human interpretable output, but suffers the poor lexical coverage problem. Gildea and Jurafsky (2002) show that clustering-based approach has better coverage than WordNet-based approach. Brockmann and Lapata (2003) find out that sophisticated WordNet-based methods do not always outperform simple frequency-based methods.

### 2.2 Distributional Models without WordNet

Alternatively, Rooth et al. (1999) propose an EM-based *clustering* smooth for SP. The key idea is to use the latent clusterings to take the place of WordNet semantic classes. Where the latent clusterings are automatically derived from distributional data based on EM algorithm. Recently, more sophisticated methods are innovated for SP based on topic models, where the *latent variables* (topics) take the place of semantic classes and distributional clusterings (Séaghdha, 2010; Ritter et al., 2010).

Without introducing semantic classes and latent variables, Keller and Lapata (2003) use the web to obtain frequencies for unseen bigrams smooth. Pantel et al. (2007) apply a collection of rules to filter out incorrect inferences for SP. Specifically, Dagan et al. (1999) introduce a general similarity-based model for word co-occurrence probabilities, which can be interpreted for SP. Similarly, Erk et al. propose an argument-oriented similarity model based on semantic or syntactic vector spaces (Erk,

2007; Erk et al., 2010). They compare several similarity functions and weighting functions in their model. Furthermore, instead of employing various similarity functions, Bergsma et al. (2008) propose a discriminative approach to learn the weights between the predicates, based on the *verb-noun* co-occurrences and other kinds of features.

Random walk model falls into the non-class based distributional approach. Previous literatures have fully studied the selection of distance or similarity functions to find out similar predicates and arguments (Dagan et al., 1999; Erk et al., 2010), or learn the weights between the predicates (Bergsma et al., 2008). Instead, we put effort in following issues: 1) how to measure SP; 2) how to transfer between predicates using random walk; 3) how to propagate the preferences for smooth. Experiments show these issues are important for SP and they should be addressed properly to achieve better results.

### 3 RSP: A Random Walk Model for SP

In this section, we briefly introduce how to address SP using random walk. We propose a framework of RSP with four components (functions). Each of them are flexible to be configured. In summary, Algorithm 1 describes the overall process.

---

#### Algorithm 1 RSP: Random walk model for SP

---

**Require:** Init bipartite graph  $G$  with raw counts

- 1: // Ranking on the bipartite graph  $G$ ;
  - 2:  $R = \Psi(G)$ ; // ranking function
  - 3: // Project  $R$  to monopartite graph  $D$
  - 4:  $D = \Phi(R)$ ; // distance function
  - 5: // Transform  $D$  to stochastic matrix  $P$
  - 6:  $P = \Delta(D)$ ; // probability function
  - 7: // Get the convergence  $\tilde{P}$
  - 8:  $\tilde{P} = \sum_{t=1}^{\infty} \frac{(dP)^t}{|(dP)^t|} = dP(I - dP)^{-1}$ ;
  - 9: **return** Smoothed bipartite graph  $\tilde{R}$
  - 10:  $\tilde{R} = \tilde{P} * R$ ; // propagation function
- 

**Bipartite Graph Construction:** For a given relation  $r$ , the observed predicate-argument pairs can be represented by a bipartite graph  $G=(X, Y, E)$ . Where  $X=\{q_1, q_2, \dots, q_m\}$  are the  $m$  predicates, and  $Y=\{a_1, a_2, \dots, a_n\}$  are the  $n$  arguments. We initiate the links  $E$  with the raw co-occurrence counts of seen predicate-argument pairs in a given generalization data. We represent the graph by an adjacency matrix with rows representing predicates and columns as arguments. For

convenience, we use indices  $i, j$  to represent predicates  $q_i, q_j$ , and  $k, l$  for arguments  $a_k, a_l$ .

We employ a preference *ranking function*  $\Psi$  to measure the SP between the predicates and arguments. It transforms  $G$  to a corresponding bipartite graph  $R$ , with links representing the strength of SP. Each row of the adjacency matrix  $R$  denotes the predicate vector  $\vec{q}_i$  or  $\vec{q}_j$ . We discuss the selection of  $\Psi$  in section 4.1.

$$\Psi := G \mapsto R \quad (1)$$

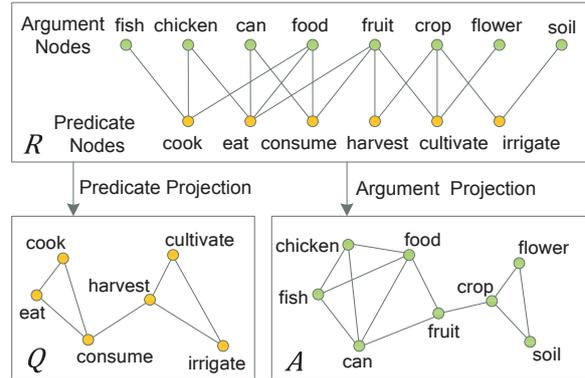


Figure 1: Illustration of  $(R)$  the bipartite graph of the *verb-dobj-noun* relation,  $(Q)$  the predicate-projection monopartite graph, and  $(A)$  the argument-projection monopartite graph.

**Monopartite Graph Projection:** In order to conduct random walk on the graph, we project the bipartite graph  $R$  onto a monopartite graph  $Q=(X, E)$  between the predicates, or  $A=(Y, E)$  between the arguments (Zhou et al., 2007). Figure 1 illustrates the intuition of the projection. The links in  $Q$  represent the indirect connects between the predicates in  $R$ . Two predicates are connected in  $Q$  if they share at least one common neighbor argument in  $R$ . The weight of the links in  $Q$  could be set by arbitrary distance measures. We refer  $D$  as an instance of the projection  $Q$  by a given *distance function*  $\Phi$ .

$$\Phi := R \mapsto D \quad (2)$$

**Stochastic Walking Strategy:** We introduce a *probability function*  $\Delta$  to transform the predicate distances  $D$  into transition probabilities  $P$ . Where  $P$  is a stochastic matrix, with each element  $p_{ij}$  represents the transition probability from predicate  $q_i$  to  $q_j$ . Generally speaking, nearby predicates gain higher probabilities to be visited, while distant predicates will be penalized.

$$\Delta := D \mapsto P \quad (3)$$

Follow Equation 4, we aggregate over all orders of the transition probabilities  $P$  as the final stationary probabilities  $\tilde{P}$ . According to the *Perron-Frobenius* theory, one can verify that it converges to  $dP(I - dP)^{-1}$  when  $P$  is non-negative and regular matrix (Li et al., 2009). Where  $t$  represents the orders: the length of the path between two nodes in terms of edges. The damp factor  $d \in (0, 1)$ , and its value mainly depends on the data sparsity level. Typically  $d$  prefers small values such as 0.005. It means higher order transitions are much less reliable than lower orders (Liben-Nowell and Kleinberg, 2007).

$$\tilde{P} = \sum_{t=1}^{\infty} \frac{(dP)^t}{|(dP)^t|} = dP(I - dP)^{-1} \quad (4)$$

**Preference Propagation:** in Equation 5, we combine the converged transition probabilities  $\tilde{P}$  with the measured preferences  $R$  as the *propagation function*: 1) for a given predicate, firstly it transfers to all nearby predicates with designed probabilities; 2) then it sums over the arguments preferred by these predicates with quantified scores to get smoothed  $\tilde{R}$ . We further describe its configuration details in Section 4.4 and Equation 12 with two propagation modes.

$$\tilde{R} = \tilde{P} * R \quad (5)$$

## 4 Model Configurations

### 4.1 Preference Ranking: Measure the Selectional Preferences

In collaborative filtering, usually there are *explicit* and *scaled* user ratings on their item preferences. For instance, a user ratings a movie with a score  $\in [0, 10]$  on IMDB site. But in SP, the preferences between the predicates and arguments are implicit: their co-occurrence counts follow the power law distribution and vary greatly.

Therefore, we employ a ranking function  $\Psi$  to measure the SP of the *seen* predicate-argument pairs. We suppose this could bring at least two benefits: 1) a proper measure on the preferences can make the discovering of nearby predicates with similar preferences to be more accurate; 2) while propagation, we propagate the scored preferences, rather than the raw counts or conditional probabilities, which could be more proper and agree with the nature of SP smooth. We denote  $\text{SelPref}(q, a)$  as  $\text{Pr}(q, a)$  for short.

$$\text{SelPref}(q, a) = \Psi(q, a) \quad (6)$$

Previous literatures have well studied on various smooth models for SP. However, they vary greatly on the measure of preferences. It is still not clear how to do this best. Lapata et al. investigate the correlations between the co-occurrence counts (CT)  $c(q, a)$ , or smoothed counts with the human plausibility judgements (Lapata et al., 1999; Lapata et al., 2001). Some introduce conditional probability (CP)  $p(a|q)$  for the decision of preference judgements (Chambers and Jurafsky, 2010; Erk et al., 2010; Séaghdha, 2010). Meanwhile, the pointwise mutual information (MI) is also employed by many researchers to filter out incorrect inferences (Pantel et al., 2007; Bergsma et al., 2008).

$$\begin{aligned} \Psi_{CT} &= c(q, a) & \Psi_{MI} &= \log \frac{p(q, a)}{p(q)p(a)} \\ \Psi_{CP} &= \frac{c(q, a)}{c(q, *)} & \Psi_{TD} &= c(q, a) \log \left( \frac{m}{|a|} \right) \end{aligned} \quad (7)$$

In this paper, we present an adjusted ranking function (AR) in Equation 8 to measure the SP of seen predicate-argument pairs. Intuitively, it measures the preferences by combining both the *popularity* and *association*, with parameters control the uncertainty of the trade-off between the two. We define the popularity as the joint probability  $p(q, a)$  based on MLE, and the association as MI. This is potentially similar to the process of human plausibility judgements. One may judge the plausibility of a predicate-argument collocation from two sides: 1) if it has enough evidences and commonly to be seen; 2) if it has strong association according to the cognition based on kinds of background knowledge. This metric is also similar to the TF-IDF (TD) used in information retrieval.

$$\begin{aligned} \Psi_{AR}(q, a) &= p(q, a)^{\alpha_1} \left( \frac{p(q, a)}{p(q)p(a)} \right)^{\alpha_2} \\ s.t. \quad &\alpha_1, \alpha_2 \in [0, 1] \end{aligned} \quad (8)$$

We verify if a metric is better by two tasks: 1) how well it correlates with human plausibility judgements; 2) how well it helps with the smooth inference to disambiguate plausible and implausible instances. We conduct empirical experiments on these issues in Section 5.3 and Section 5.4.

### 4.2 Distance Function: Projection of the Monopartite Graph

In Equation 9, the distance function  $\Phi$  is used to discover nearby predicates with distance  $d_{ij}$ . It weights the links on the monopartite graph  $Q$ . It

guides the walker to transfer between predicates. We calculate  $\Phi$  based on the vectors  $\vec{q}_i, \vec{q}_j$  represented by the measured preferences in  $R$ .

$$d_{ij} = \Phi(\vec{q}_i, \vec{q}_j) \quad (9)$$

Where  $\Phi$  can be distance functions such as Euclidean (norm) distance or Kullback-Leibler divergence (KL) etc., or one *minus* the similarity functions such as Jaccard and Cosine etc. The selection of distributional functions has been fully studied by previous work (Lee, 1999; Erk et al., 2010). In this paper, we do not focus on this issue due to page limits. We simply use the *Cosine* function:

$$\Phi_{\text{cosine}}(\vec{q}_i, \vec{q}_j) = 1 - \frac{\vec{q}_i \cdot \vec{q}_j}{\|\vec{q}_i\| \|\vec{q}_j\|} \quad (10)$$

### 4.3 Probability Function: the Walk Strategy

We define the probability function  $\Delta$  as Equation 11. Where the transition probability  $p(q_j|q_i)$  in  $P$  is defined as a function of the distance  $d_{ij}$  with a parameter  $\delta$ . Intuitively, it means in a given walk step, a predicate  $q_j$  which is far away from  $q_i$  will get much less probability to be visited, and  $q_i$  has high probabilities to start walk from itself and its nearby predicates to pursue good precision. Once we get the transition matrix  $P$ , we can compute  $\tilde{P}$  according to Equation 4.

$$p(q_j|q_i) = \Delta(d_{ij}) = \frac{(1 - d_{ij})^\delta}{Z(q_i)} \quad (11)$$

*s.t.*  $\delta \geq 0, \quad d_{ij} \in [0, 1]$

Where the parameter  $\delta$  is used to control the balance of nearby and distant predicates.  $Z(q_i)$  is the normalize factor. Typically,  $\delta$  around 2 can produce good enough results in most cases. We verify the settings of  $\delta$  in section 5.3.2.

### 4.4 Propagation Function

The propagation function in Equation 5 is represented by the matrix form. It can be expanded and rewritten as Equation 12. Where  $\tilde{p}(q_j|q_i)$  is the converged transition probability from predicate  $q_i$  to  $q_j$ .  $\Pr(a_k, q_j)$  is the measured preference of predicate  $q_j$  with argument  $a_k$ .

$$\tilde{\Pr}(a_k, q_i) = \sum_{j=1}^m \tilde{p}(q_j|q_i) \cdot \Pr(a_k, q_j) \quad (12)$$

We employ two propagation modes (PropMode) for the preference propagation function. One is

'CP' mode. In this mode, we always set  $\Pr(q, a)$  as the conditional probability  $p(a|q)$  for the propagation function, despite what  $\Psi$  is used for the distance function. This mode is similar to previous methods (Dagan et al., 1999; Keller and Lapata, 2003; Bergsma et al., 2008). The other is 'PP' mode. We set ranking function  $\Psi = \Pr(q, a)$  always to be the same in both the distance function and the propagation function. That means what we propagated is the designed and scored preferences. This could be more proper and agree with the nature of SP smooth. We show the improvement of this extension in section 5.3.1.

## 5 Experiments

### 5.1 Data Set

**Generalization Data:** We parsed the Agence France-Presse (AFP) and New York Times (NYT) sections of the LDC English Gigaword corpora (Parker et al., 2011), each from year 2001-2010. The parser is provided by the Stanford CoreNLP package<sup>1</sup>. We filter out all tokens containing non-alphabetic characters, collect the *<verb-dobj-noun>* triples from the syntactically analyzed data. Predicates (verbs) whose frequency lower than 30 and arguments (noun headwords) whose frequency less than 5 are excluded out. No other filters have been done. The resulting data consist of:

- **AFP:** 26, 118, 892 verb-dobj-noun observations with 1, 918, 275 distinct triples, totally 4, 771 predicates and 44, 777 arguments.
- **NYT:** 29, 149, 574 verb-dobj-noun observations with 3, 281, 391 distinct triples, totally 5, 782 predicates and 57, 480 arguments.

**Test Data:** For pseudo-disambiguation, we employ Penn TreeBank-3 (**PTB**) as the test data (Marcus et al., 1999)<sup>2</sup>. We collect the 36, 400 manually annotated *verb-dobj-noun* dependencies (with 23, 553 distinct ones) from PTB. We keep dependencies whose predicates and arguments are seen in the generalization data. We randomly select 20% of these dependencies as the test set. We split the test set equally into two parts: one as the *development* set and the other as the *final* test set.

**Human Plausibility Judgements Data:** We employ two human plausibility judgements data

<sup>1</sup><http://nlp.stanford.edu/software/corenlp.shtml>

<sup>2</sup>PTB includes 2, 499 stories from the Wall Street Journal (WSJ). It is different with our two generalization data.

for the correlation evaluation. In each they collect a set of predicate-argument pairs, and annotate with two kinds of human ratings: one for an argument takes the role as the *patient* of a predicate, and the other for the argument as the *agent*. The rating values are between 1 and 7: e.g. they assign *hunter-subj-shoot* with a rating 6.9 but 2.8 for *shoot-dobj-hunter*.

- **PBP**: Padó et al. (2007) develop a set of human plausibility ratings on the basis of the Penn TreeBank and FrameNet respectively. We refer PBP as their 212 patient ratings from the Penn TreeBank.
- **MRP**: This data are originally contributed by McRae et al. (1998). We use all their 723 *patient-nn* ratings.

Without explicit explanation, we remove all the selected PTB tests and human plausibility pairs from AFP and NYT to treat them unseen.

## 5.2 Comparison Methods

Since RSP falls into the unsupervised distributional approach, we compare it with previous similarity-based methods and unsupervised generative topic model<sup>3</sup>.

**Erk** et al. (Erk, 2007; Erk et al., 2010) are the pioneers to address SP using similarity-based method. For a given  $(q, a)$  in relation  $r$ , the model sums over the similarities between  $a$  and the seen headwords  $a' \in Seen(q, r)$ . They investigated several similarity functions  $sim(a, a')$  such as Jaccard, Cosine, Lin, and nGCM etc., and different weighting functions  $wt_{q,r}(a')$ .

$$S(q, r, a) = \sum_{a'} \frac{wt_{q,r}(a')}{Z_{q,r}} \cdot sim(a, a') \quad (13)$$

For comparison, we suppose the primary corpus and generalization corpus in their model to be the same. We set the similarity function of their model as nGCM, use both the **FREQ** and **DISCR** weighting functions. The vector space is in **SYN-PRIMARY** setting with 2,000 basis elements.

**Dagan** et al. (1999) propose state-of-the-art similarity based model for word co-occurrence probabilities. Though it is not intended for SP, but it can be interpreted and rewritten for SP as:

$$\Pr(a|q) = \sum_{q' \in Simset(q)} \frac{sim(q, q')}{Z(q)} p(a|q') \quad (14)$$

<sup>3</sup>The implementation of RSP and listed previous methods are available at <https://github.com/ZhenhuaTian/RSP>

They use the  $k$ -closest nearbys as  $Simset(q)$ , with a parameter  $\beta$  to revise the similarity function. For comparison, we use the Jensen-Shannon divergence (Lin, 1991) which shows the best performance in their work as  $sim(q, q')$ , and optimize the settings of  $k$  and  $\beta$  in our experiments.

**LDA-SP**: Another kind of sophisticated unsupervised approaches for SP are latent variable models based on Latent Dirichlet Allocation (LDA). Ó Séaghdha (2010) applies topic models for the SP induction with three variations: LDA, Rooth-LDA, and Dual-LDA; Ritter et al. (2010) focus on inferring latent topics and their distributions over multiple arguments and relations (e.g., the subject and direct object of a verb).

In this work, we compare with Ó Séaghdha’s original LDA approach to SP. We use the Matlab Topic Modeling Toolbox<sup>4</sup> for the inference of latent topics. The hyper parameters are set as suggested  $\alpha=50/T$  and  $\beta=200/n$ , where  $T$  is the number of topics and  $n$  is the number of arguments. We test  $T=100, 200, 300$ , each with 1,000 iterations of Gibbs sampling.

## 5.3 Pseudo-Disambiguation

*Pseudo-disambiguation* has been used for SP evaluation by many researchers (Rooth et al., 1999; Erk, 2007; Bergsma et al., 2008; Chambers and Jurafsky, 2010; Ritter et al., 2010). First the system removes a portion of seen predicate-argument pairs from the generalization data to treat them as unseen positive tests  $(q, a^+)$ . Then it introduces confounder selection to create a *pseudo* negative test  $(q, a^-)$  for each positive  $(q, a^+)$ . Finally it evaluates a SP model by how well the model disambiguates these positive and negative tests.

**Confounder Selection**: for a given  $(q, a^+)$ , the system selects an argument  $a'$  from the argument vocabulary. Then by ensure  $(q, a')$  is *unseen* in the generalization data, it treats  $a'$  as pseudo  $a^-$ . This process guarantees that  $(q, a^-)$  to be negative in real case with very high probability. Previous work have made advances on confounder selection with random, bucket and nearest confounders. Random confounder (RND) most closes to the realistic case; While nearest confounder (NER) is reproducible and it avoids frequency bias (Chambers and Jurafsky, 2010).

In this work, we employ both RND and NER confounders: 1) for RND, we randomly select

<sup>4</sup>[psiexp.ss.uci.edu/research/programs\\_data/toolbox.htm](https://psiexp.ss.uci.edu/research/programs_data/toolbox.htm)

confounders according to the occurrence probability of arguments. We sample confounders on both the development and final test data with 100 iterations. 2) for NER, firstly we sort the arguments by their frequency. Then we select the nearest confounders with two iterations. One iteration selects the confounder whose frequency is more than or equal to  $a^+$ , and the other iteration with frequency lower than or equal to  $a^+$ .

**Evaluation Metric:** we evaluate performance on both the *pairwise* and *pointwise* settings:

1) On pairwise setting, we combine corresponding  $(q, a^+, a^-)$  together as test instances. The performance is evaluated based on the accuracy (ACC) metric. It computes the portion of test instances  $(q, a^+, a^-)$  which correctly predicted by the smooth model with  $\text{score}(q, a^+) > \text{score}(q, a^-)$ . We weight each instance equally for **macroACC**, and weight each by the frequency of the positive pair  $(q, a^+)$  for **microACC**.

2) On pointwise setting, we use each positive test  $(q, a^+)$  or negative test  $(q, a^-)$  as test instances independently. We treat it as a binary classification task, and evaluate using the standard area-under-the-curve (AUC) metric. This metric is firstly employed for the SP evaluation by Ritter et al (2010). For **macroAUC**, we weight each instance equally; for **microAUC**, we weight each by its argument frequency (Bergsma et al., 2008).

**Parameters Tuning:** The parameters are tuned on the PTB *development* set, using AFP as the generalization data. We report the overall performance on the *final* test set. While using NYT as the generalization data, we hold the same parameter settings as AFP to ensure the results are robust. Note that indeed the parameter settings would vary among different generalization and test data.

### 5.3.1 Verify Ranking Function and Propagation Method

This experiment is conducted on the PTB *development* set with RND confounders. We use AFP and NYT as the generalization data. For comparison, we set the distance function  $\Phi$  as *Cosine*, with default  $d=0.005$ , and  $\delta=1$ .

In Table 1, the evaluation metric is Accuracy. The first 4 rows are the results of 'CP' PropMode, and the latter 3 rows are the 'PP' PropMode. With respect to the ranking function  $\Psi$ , CP performs the worst as it considers only the popularity rather than association. The heavy bias on frequent predicates and arguments has two major drawbacks: a)

The computation of predicate distances would rely much more on frequent arguments, rather than those arguments they preferred; b) While propagation, it may bias more on frequent arguments, too. Even these frequent arguments are less preferred and not proper to be propagated.

Crit.	AFP		NYT	
	macro	micro	macro	micro
$\Psi_{CP}$	71.7	76.7	78.2	81.2
$\Psi_{MI}$	70.9	75.8	79.1	81.8
$\Psi_{TD}$	73.4	78.2	80.9	83.4
$\Psi_{AR}$	72.9	77.8	81.0	83.5
$\Psi_{MI}$	76.8	80.6	81.9	83.8
$\Psi_{TD}$	74.4	79.1	81.8	84.2
$\Psi_{AR}$	82.5	85.2	87.7	88.6

Table 1: Comparing different ranking functions.

For MI, it biases infrequent arguments with strong association, without regarding to the popular arguments with more evidences. Furthermore, the generalization data is automatically parsed and kind of noisy, especially on infrequent predicates and arguments. The noises could yield unreliable estimations and decrease the performance. For TD, it outperforms MI method on 'CP' PropMode, but it not always outperforms MI on 'PP' PropMode. It is no surprise to find out the adjusted ranking AR achieves better results on both AFP and NYT data, with  $\alpha_1=0.2$  and  $\alpha_2=0.6$ . Finally, it shows the 'PP' mode, which propagating the designed preference scores, gains significantly better performance as discussed in Section 4.4.

### 5.3.2 Verify $\delta$ of the Probability Function

This experiment is conducted on the PTB *development* tests with both RND and NER confounders. The generalization data is AFP.

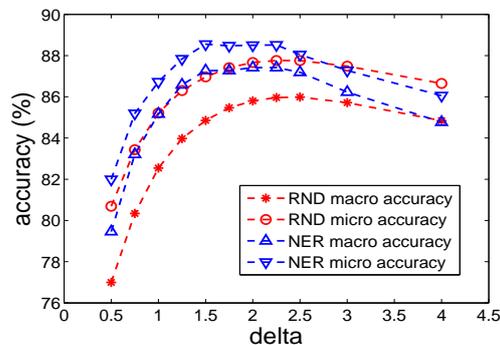


Figure 2: Performance variation on different  $\delta$ .

Criterion	AFP				NYT			
	RND		NER		RND		NER	
	macro	micro	macro	micro	macro	micro	macro	micro
Erk et al. <i>FREQ</i>	73.7	73.6	73.9	73.6	68.3	68.4	63.8	63.0
Erk et al. <i>DISCR</i>	76.0	78.3	79.1	78.1	83.3	84.2	82.4	82.6
Dagan et al.	80.6	82.8	84.7	85.0	87.0	87.6	86.9	87.3
LDA-SP	82.0	83.5	83.7	82.9	89.1	89.0	87.9	87.8
RSP <sub>naive</sub>	72.6	76.4	79.4	81.1	78.5	80.4	74.8	78.0
+Rank	74.0	77.7	83.5	85.2	81.4	83.1	84.5	86.9
+Rank+PP	83.5	85.2	87.2	87.0	88.2	88.2	88.0	88.3
+Rank+PP+Delta	<b>86.2</b>	<b>87.3</b>	<b>88.4</b>	<b>88.1</b>	<b>90.6</b>	<b>90.1</b>	<b>91.1</b>	<b>89.3</b>

Table 2: Pseudo-disambiguation results of different smooth models. Macro and micro Accuracy.

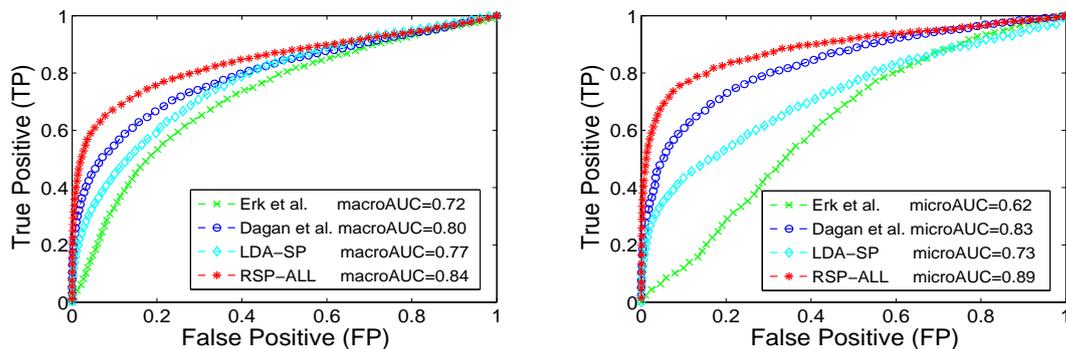


Figure 3: Marco and micro ROC curves of different smooth models.

We set the ranking function  $\Psi$  as AR (with tuned  $\alpha_1=0.2$  and  $\alpha_2=0.6$ ), the distance function  $\Phi$  as *Cosine*, default  $d=0.005$ , and we restrict  $\delta \in [0.5, 4]$ . Figure 2 shows  $\delta$  has significant impact on the performance. Starting from  $\delta=0.5$ , the system gains better performance while  $\delta$  increasing. It achieves good results around  $\delta=2$ . This means for a given predicate, the penalty on its distant predicates helps to get more accurate smooth. The performance will drop if  $\delta$  becomes too big. This means closest predicates are useful for smooth. It is not better to penalize them heavily.

### 5.3.3 Overall Performance

Finally we compare the overall performance of different models. We report the results on the PTB *final* test set, with RND and NER confounders.

Table 2 shows the overall performance on Accuracy metric. Among previous methods in the first 4 rows, LDA-SP performs the best in most cases. In the last 4 rows, RSP<sub>naive</sub> means both the ranking function and PropMode are set as 'CP' and  $\delta=1$ . This configuration yields poor performance. Iteratively, by employing the adjusted

ranking function, smoothing with preference propagation method, and revising the probability function with the parameter  $\delta$ , RSP outperforms all previous methods. The parameter settings of RSP-All are  $\alpha_1=0.2$ ,  $\alpha_2=0.6$ ,  $\delta=1.75$  and  $d=0.005$ .

Figure 3 show the macro (left) and micro (right) receiver-operating-characteristic (ROC) curves of different models, using AFP as the generalization data and RND confounders. For each kind of previous methods, we show the best AUC they achieved. RASP-All still performs the best on the terms of AUC metric, achieving macroAUC at 84% and microAUC at 89%. We also verified the AUC metric using NYT as the generalization data. The results are similar to the AFP data. It is also interesting to find out that the ACC metric is not always bring into correspondence with the AUC metric. The difference mainly raise on the pointwise and pairwise test settings of pseudo-disambiguation.

### 5.4 Human Plausibility Judgements

We conduct empirical studies on the correlations between different *preference ranking func-*

Criterion	AFP				NYT			
	Spearman's $\rho$		Kendall's $\tau$		Spearman's $\rho$		Kendall's $\tau$	
	PBP	MRP	PBP	MRP	PBP	MRP	PBP	MRP
CT	0.49	0.36	0.37	0.28	0.54	0.44	0.41	0.34
CP	0.47	0.39	0.35	0.30	0.51	0.48	0.39	0.37
MI	0.56	0.39	0.43	<b>0.31</b>	0.54	0.49	0.41	0.38
TD	0.53	0.36	0.39	0.28	0.56	0.45	0.42	0.34
AR	<b>0.58</b>	<b>0.40</b>	<b>0.44</b>	<b>0.31</b>	<b>0.58</b>	<b>0.50</b>	<b>0.44</b>	<b>0.39</b>
Erk et al. <i>FREQ</i>	0.30	0.08	0.22	0.06	0.25	0.09	0.18	0.06
Erk et al. <i>DISCR</i>	0.06	0.21	0.04	0.15	0.16	0.23	0.11	0.16
Dagan et al.	0.32	0.24	0.24	0.18	0.46	0.29	0.34	0.21
LDA-SP	0.31	<b>0.32</b>	0.23	<b>0.23</b>	0.38	<b>0.38</b>	0.28	<b>0.28</b>
LDA-SP <sub>+Bayes</sub>	0.39	0.25	0.30	0.18	0.40	0.32	0.30	0.23
RSP-All	<b>0.46</b>	0.31	<b>0.34</b>	<b>0.23</b>	<b>0.53</b>	<b>0.38</b>	<b>0.40</b>	<b>0.28</b>

Table 3: Correlation results on the human plausibility judgements data.

tions and human ratings. Follow Lapata et al. (2001), we first collect the co-occurrence counts of predicate-argument pairs in the human plausibility data from AFP and NYT (before removing them as unseen pairs). Then we score them with different ranking functions (described in Section 4.1) based on MLE. Inspired by Erk et al. (2010), we do not suppose linear correlations between the estimated scores and human ratings. We use the Spearman's  $\rho$  and Kendall's  $\tau$  **rank correlation** coefficient.

We also compare the correlations between the *smoothed scores* of different models with human ratings. With respect to **upper bounds**, Padó et al. (2007) suggest that the typical agreement of human participants is around a correlation of 0.7 on their plausibility data. We hold that automatic models of plausibility can not be expected to surpass this upper bound.

In Table 3, all coefficients are verified at significant level  $p < 0.01$ . The first 5 rows are the correlations between the preference ranking functions and human ratings based on MLE. On both the PBP and MRP data, the proposed AR metric better correlates with human ratings than others, with  $\alpha_2 > 0.5$  and  $\alpha_1$  around  $[0.2, 0.35]$ . The latter 6 rows are the results of smooth models. It shows LDA-SP performs good correlation with human ratings, where LDA-SP<sub>+Bayes</sub> refers to the Bayes prediction method of Ritter et al. (2010). RSP model gains the best correlation on the two plausibility data in most cases, where the parameter settings are the same as pseudo-disambiguation.

## 6 Conclusions and Future Work

In this work we present a random walk approach to SP. Experiments show it is efficient and effective to address data sparsity for SP. It is also flexible to be applied to new data. We find out that a proper measure on SP between the predicates and arguments is important for SP. It helps with the discovering of nearby predicates and it makes the preference propagation to be more accurate. Another issue is that it is not good enough to directly apply the similarity or distance functions for smooth. Potential future work including but not limited to follows: investigate argument-oriented and personalized random walk, extend the model in heterogeneous network with multiple link types, discover soft clusters using random walk for semantic induction, and combine it with discriminative learning approach etc.

## Acknowledgments

The research is supported in part by the National High Technology Research and Development Program 863 of China under Grant No.2012AA011003; Key Projects in the National Science and Technology Pillar Program under Grant No.2011BAK08B02; Chinese Government Graduate Student Overseas Study Program sponsored by the China Scholarship Council (CSC). We also gratefully acknowledge the anonymous reviewers for their helpful comments.

## References

- Eneko Agirre and David Martinez. 2001. Learning class-to-class selectional preferences. In *Proceedings of the 2001 workshop on Computational Natural Language Learning*.
- Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *EMNLP*.
- Carsten Brockmann and Mirella Lapata. 2003. Evaluating and combining approaches to selectional preference acquisition. In *EACL*.
- Nathanael Chambers and Dan Jurafsky. 2010. Improving the use of pseudo-words for evaluating selectional preferences. In *ACL*.
- Massimiliano Ciaramita and Mark Johnson. 2000. Explaining away ambiguity: Learning verb selectional preference with bayesian networks. In *COLING*.
- Stephen Clark and David J. Weir. 2002. Class-based probability estimation using a semantic hierarchy. *Computational Linguistics*, 28(2):187–206.
- Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. 1999. Similarity-Based Models of Word Cooccurrence Probabilities. *Machine Learning*, 34:43–69.
- Katrin Erk, Sebastian Padó, and Ulrike Padó. 2010. A flexible, corpus-driven model of regular and inverse selectional preferences. *Computational Linguistics*, 36(4):723–763.
- Katrin Erk. 2007. A simple, similarity-based model for selectional preferences. In *ACL*.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.
- Jerrold J. Katz and Jerry A. Fodor. 1963. The structure of a semantic theory. *Language*, 39(2):170–210.
- Frank Keller and Mirella Lapata. 2003. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3):459–484.
- Maria Lapata, Scott McDonald, and Frank Keller. 1999. Determinants of adjective-noun plausibility. In *EACL*, pages 30–36. Association for Computational Linguistics.
- Maria Lapata, Frank Keller, and Scott McDonald. 2001. Evaluating smoothing algorithms against plausibility judgements. In *ACL*, pages 354–361. Association for Computational Linguistics.
- Lillian Lee. 1999. Measures of distributional similarity. In *ACL*, pages 25–32, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hang Li and Naoki Abe. 1998. Generalizing case frames using a thesaurus and the mdl principle. *Computational linguistics*, 24(2):217–244.
- Ming Li, Benjamin M Dias, Ian Jarman, Wael El-Dereedy, and Paulo JG Lisboa. 2009. Grocery shopping recommendations based on basket-sensitive random walk. In *SIGKDD*, pages 1215–1224. ACM.
- David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031.
- Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.
- Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3.
- Diana McCarthy, Sriram Venkatapathy, and Aravind K. Joshi. 2007. Detecting compositionality of verb-object combinations using selectional preferences. In *EMNLP-CoNLL*.
- Ken McRae, Michael J. Spivey-Knowlton, and Michael K. Tanenhaus. 1998. Modeling the influence of thematic fit (and other constraints) in on-line sentence comprehension. *Journal of Memory and Language*, 38(3):283–312.
- Sebastian Padó, Ulrike Padó, and Katrin Erk. 2007. Flexible, corpus-based modelling of human plausibility judgements. In *EMNLP/CoNLL*, volume 7.
- Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. Isp: Learning inferential selectional preferences. In *NAACL-HLT*.
- Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition.
- Philip Resnik. 1993. Selection and information: a class-based approach to lexical relationships. *IRCS Technical Reports Series*.
- Philip Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, 61(1):127–159.
- Philip Resnik. 1997. Selectional preference and sense disambiguation. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How*. Washington, DC.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *ACL*.
- Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a semantically annotated lexicon via em-based clustering. In *ACL*.
- Diarmuid Ó Séaghdha. 2010. Latent variable models of selectional preference. In *ACL*.

- Irena Spasić and Sophia Ananiadou. 2004. Using automatically learnt verb selectional preferences for classification of biomedical terms. *Journal of Biomedical Informatics*, 37(6):483–497.
- Kristina Toutanova, Christopher D. Manning, Dan Flickinger, and Stephan Oepen. 2005. Stochastic hpsg parse disambiguation using the redwoods corpus. *Research on Language & Computation*, 3(1):83–105.
- Yorick Wilks. 1973. Preference semantics. Technical report, DTIC Document.
- Hilmi Yildirim and Mukkai S. Krishnamoorthy. 2008. A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 131–138. ACM.
- Tao Zhou, Jie Renan, Matúš Medo, and Yi-Cheng Zhang. 2007. Bipartite network projection and personal recommendation. *Physical Review E*, 76(4):046115.

# ImpAr: A Deterministic Algorithm for Implicit Semantic Role Labelling

**Egoitz Laparra**

IXA Group  
University of the Basque Country  
San Sebastian, Spain  
egoitz.laparra@ehu.es

**German Rigau**

IXA Group  
University of the Basque Country  
San Sebastian, Spain  
german.rigau@ehu.es

## Abstract

This paper presents a novel deterministic algorithm for implicit Semantic Role Labeling. The system exploits a very simple but relevant discursive property, the argument coherence over different instances of a predicate. The algorithm solves the implicit arguments sequentially, exploiting not only explicit but also the implicit arguments previously solved. In addition, we empirically demonstrate that the algorithm obtains very competitive and robust performances with respect to supervised approaches that require large amounts of costly training data.

## 1 Introduction

Traditionally, Semantic Role Labeling (SRL) systems have focused in searching the fillers of those explicit roles appearing within sentence boundaries (Gildea and Jurafsky, 2000, 2002; Carreras and Màrquez, 2005; Surdeanu et al., 2008; Hajič et al., 2009). These systems limited their search-space to the elements that share a syntactical relation with the predicate. However, when the participants of a predicate are implicit this approach obtains incomplete predicative structures with null arguments. The following example includes the gold-standard annotations for a traditional SRL process:

- (1) [*arg0* The network] had been expected to have [*np losses*] [*arg1* of as much as \$20 million] [*arg3* on baseball this year]. It isn't clear how much those [*np losses*] may widen because of the short Series.

The previous analysis includes annotations for the nominal predicate **loss** based on the NomBank structure (Meyers et al., 2004). In this case the annotator identifies, in the first sentence, the arguments *arg0*, the entity losing something, *arg1*, the

thing lost, and *arg3*, the source of that loss. However, in the second sentence there is another instance of the same predicate, **loss**, but in this case no argument has been associated with it. Traditional SRL systems facing this type of examples are not able to fill the arguments of a predicate because their fillers are not in the same sentence of the predicate. Moreover, these systems also let unfilled arguments occurring in the same sentence, like in the following example:

- (2) Quest Medical Inc said it adopted [*arg1* a shareholders' rights] [*np plan*] in which rights to purchase shares of common stock will be distributed as a dividend to shareholders of record as of Oct 23.

For the predicate **plan** in the previous sentence, a traditional SRL process only returns the filler for the argument *arg1*, the theme of the plan.

However, in both examples, a reader could easily infer the missing arguments from the surrounding context of the predicate, and determine that in (1) both instances of the predicate share the same arguments and in (2) the missing argument corresponds to the subject of the verb that dominates the predicate, *Quest Medical Inc*. Obviously, this additional annotations could contribute positively to its semantic analysis. In fact, Gerber and Chai (2010) pointed out that implicit arguments can increase the coverage of argument structures in NomBank by 71%. However, current automatic systems require large amounts of manually annotated training data for each predicate. The effort required for this manual annotation explains the absence of generally applicable tools. This problem has become a main concern for many NLP tasks. This fact explains a new trend to develop accurate unsupervised systems that exploit simple but robust linguistic principles (Raghuathan et al., 2010).

In this work, we study the coherence of the predicate and argument realization in discourse. In particular, we have followed a similar approach to

the one proposed by Dahl et al. (1987) who filled the arguments of anaphoric mentions of nominal predicates using previous mentions of the same predicate. We present an extension of this idea assuming that in a coherent document the different occurrences of a predicate, including both verbal and nominal forms, tend to be mentions of the same event, and thus, they share the same argument fillers. Following this approach, we have developed a deterministic algorithm that obtains competitive results with respect to supervised methods. That is, our system can be applied to any predicate without training data.

The main contributions of this work are the following:

- We empirically prove that there exists a strong discourse relationship between the implicit and explicit argument fillers of the same predicates.
- We propose a deterministic approach that exploits this discursive property in order to obtain the fillers of implicit arguments.
- We adapt to the implicit SRL problem a classic algorithm for pronoun resolution.
- We develop a robust algorithm, ImpAr, that obtains very competitive results with respect to existing supervised systems. We release an open source prototype implementing this algorithm<sup>1</sup>.

The paper is structured as follows. Section 2 discusses the related work. Section 3 presents in detail the data used in our experiments. Section 4 describes our algorithm for implicit argument resolution. Section 5 presents some experiments we have carried out to test the algorithm. Section 6 discusses the results obtained. Finally, section 7 offers some concluding remarks and presents some future research lines.

## 2 Related Work

The first attempt for the automatic annotation of implicit semantic roles was proposed by Palmer et al. (1986). This work applied selectional restrictions together with coreference chains, in a very specific domain. In a similar approach, Whittemore et al. (1991) also attempted to solve implicit

arguments using some manually described semantic constraints for each thematic role they tried to cover. Another early approach was presented by Tetreault (2002). Studying another specific domain, they obtained some probabilistic relations between some roles. These early works agree that the problem is, in fact, a special case of anaphora or coreference resolution.

Recently, the task has been taken up again around two different proposals. On the one hand, Ruppenhofer et al. (2010) presented a task in SemEval-2010 that included an implicit argument identification challenge based on FrameNet (Baker et al., 1998). The corpus for this task consisted in some novel chapters. They covered a wide variety of nominal and verbal predicates, each one having only a small number of instances. Only two systems were presented for this sub-task obtaining quite poor results (F1 below 0,02). VENSES++ (Tonelli and Delmonte, 2010) applied a rule based anaphora resolution procedure and semantic similarity between candidates and thematic roles using WordNet (Fellbaum, 1998). The system was tuned in (Tonelli and Delmonte, 2011) improving slightly its performance. SEMAFOR (Chen et al., 2010) is a supervised system that extended an existing semantic role labeler to enlarge the search window to other sentences, replacing the features defined for regular arguments with two new semantic features. Although this system obtained the best performance in the task, data sparseness strongly affected the results. Besides the two systems presented to the task, some other systems have used the same dataset and evaluation metrics. Ruppenhofer et al. (2011), Laparra and Rigau (2012), Gorinski et al. (2013) and Laparra and Rigau (2013) explore alternative linguistic and semantic strategies. These works obtained significant gains over previous approaches. Silberer and Frank (2012) adapted an entity-based coreference resolution model to extend automatically the training corpus. Exploiting this additional data, their system was able to improve previous results. Following this approach Moor et al. (2013) present a corpus of predicate-specific annotations for verbs in the FrameNet paradigm that are aligned with PropBank and VerbNet.

On the other hand, Gerber and Chai (2010, 2012) studied the implicit argument resolution on NomBank. They use a set of syntactic, semantic and coreferential features to train a logistic regres-

---

<sup>1</sup><http://adimen.si.ehu.es/web/ImpAr>

sion classifier. Unlike the dataset from SemEval-2010 (Ruppenhofer et al., 2010), in this work the authors focused on a small set of ten predicates. But for those predicates, they annotated a large amount of instances in the documents from the Wall Street Journal that were already annotated for PropBank (Palmer et al., 2005) and NomBank. This allowed them to avoid the sparseness problems and generalize properly from the training set. The results of this system were far better than those obtained by the systems that faced the SemEval-2010 dataset. This work represents the deepest study so far of the features that characterize the implicit arguments<sup>2</sup>. However, many of the most important features are lexically dependent on the predicate and cannot be generalized. Thus, specific annotations are required for each new predicate to be analyzed.

All the works presented in this section agree that implicit arguments must be modeled as a particular case of coreference together with features that include lexical-semantic information, to build selectional preferences. Another common point is the fact that these works try to solve each instance of the implicit arguments independently, without taking into account the previous realizations of the same implicit argument in the document. We propose that these realizations, together with the explicit ones, must maintain a certain coherence along the document and, in consequence, the filler of an argument remains the same along the following instances of that argument until a stronger evidence indicates a change. We also propose that this feature can be exploited independently from the predicate.

### 3 Datasets

In our experiments, we have focused on the dataset developed in Gerber and Chai (2010, 2012). This dataset (hereinafter BNB which stands for "Beyond NomBank") extends existing predicate annotations for NomBank and PropBank.

BNB presented the first annotation work of implicit arguments based on PropBank and NomBank frames. This annotation was an extension of the standard training, development and testing sections of Penn TreeBank that have been typically used for SRL evaluation and were already annotated with PropBank and NomBank predicate

<sup>2</sup>Gerber and Chai (2012) includes a set of 81 different features.

structures. The authors selected a limited set of predicates. These predicates are all nominalizations of other verbal predicates, without sense ambiguity, that appear frequently in the corpus and tend to have implicit arguments associated with their instances. These constraints allowed them to model enough occurrences of each implicit argument in order to cover adequately all the possible cases appearing in a test document. For each missing argument position they went over all the preceding sentences and annotated all mentions of the filler of that argument. In tables 3 and 4 we show the list of predicates and the resulting figures of this annotation.

In this work we also use the corpus provided for the CoNLL-2008 task. These corpora cover the same BNB documents and include annotated predictions for syntactic dependencies and SuperSense labels as semantic tags. Unlike Gerber and Chai (2010, 2012) we do not use the constituent analysis from the Penn TreeBank.

## 4 ImpAr algorithm

### 4.1 Discursive coherence of predicates

Exploring the training dataset of BNB, we observed a very strong discourse effect on the implicit and explicit argument fillers of the predicates. That is, if several instances of the same predicate appear in a well-written discourse, it is very likely that they maintain the same argument fillers. This property holds when joining the different parts-of-speech of the predicates (nominal or verbal) and the explicit or implicit realizations of the argument fillers. For instance, we observed that 46% of all implicit arguments share the same filler with the previous instance of the same predicate while only 14% of them have a different filler. The remaining 40% of all implicit arguments correspond to first occurrences of their predicates. That is, these fillers can not be recovered from previous instances of their predicates.

The rationale behind this phenomena seems to be simple. When referring to different aspects of the same event, the writer of a coherent document does not repeat redundant information. They refer to previous predicate instances assuming that the reader already recalls the involved participants. That is, the filler of the different instances of a predicate argument maintain a certain discourse coherence. For instance, in example (1), all the argument positions of the second occurrence of the

predicate **loss** are missing, but they can be easily inferred from the previous instance of the same predicate.

- (1) [*arg0* The network] had been expected to have [*np losses*] [*arg1* of as much as \$20 million] [*arg3* on baseball this year]. It isn't clear how much those [*np losses*] may widen because of the short Series.

Therefore, we propose to exploit this property in order to capture correctly how the fillers of all predicate arguments evolve through a document.

Our algorithm, **ImpAr**, processes the documents sentence by sentence, assuming that sequences of the same predicate (in its nominal or verbal form) share the same argument fillers (explicit or implicit)<sup>3</sup>. Thus, for every **core** argument  $arg_n$  of a predicate, ImpAr stores its previous known filler as a **default** value. If the arguments of a predicate are explicit, they always replace default fillers previously captured. When there is no antecedent for a particular implicit argument  $arg_n$ , the algorithm tries to find in the surrounding context which participant is the most likely to be the filler according to some salience factors (see Section 4.2). For the following instances, without an explicit filler for a particular argument position, the algorithm repeats the same selection process and compares the new implicit candidate with the default one. That is, the default implicit argument of a predicate with no antecedent can change every time the algorithm finds a filler with a greater salience. A damping factor is applied to reduce the salience of distant predicates.

#### 4.2 Filling arguments without explicit antecedents

Filling the implicit arguments of a predicate has been identified as a particular case of coreference, very close to pronoun resolution (Silberer and Frank, 2012). Consequently, for those implicit arguments that have not explicit antecedents, we propose an adaptation of a classic algorithm for deterministic pronoun resolution. This component of our algorithm follows the RAP approach (Lapin and Leass, 1994). When our algorithm needs to fill an implicit predicate argument without an explicit antecedent it considers a set of candidates within a window formed by the sentence of the predicate and the two previous sentences. Then, the algorithm performs the following steps:

1. Apply two constraints to the candidate list:
  - (a) All candidates that are already explicit arguments of the predicate are ruled out.
  - (b) All candidates commanded by the predicate in the dependency tree are ruled out.
2. Select those candidates that are semantically consistent with the semantic category of the implicit argument.
3. Assign a salience score to each candidate.
4. Sort the candidates by their proximity to the predicate of the implicit argument.
5. Select the candidate with the highest salience value.

As a result, the candidate with the highest salience value is selected as the filler of the implicit argument. Thus, this filler with its corresponding salience weight will be also considered in subsequent instances of the same predicate.

Now, we explain each step in more detail using example (2). In this example,  $arg_0$  is missing for the predicate **plan**:

- (2) Quest Medical Inc said it adopted [*arg1* a shareholders' rights] [*np plan*] in which rights to purchase shares of common stock will be distributed as a dividend to shareholders of record as of Oct 23.

**Filtering.** In the first step, the algorithm filters out the candidates that are actual explicit arguments of the predicate or have a syntactic dependency with the predicate, and therefore, they are in the search space of a traditional SRL system.

In our example, the filtering process would remove [*a shareholders' rights*] because it is already the explicit argument  $arg_1$ , and [*in which rights to purchase shares of common stock will be distributed as a dividend to shareholders of record as of Oct 23*] because it is syntactically commanded by the predicate **plan**.

**Semantic consistency.** To determine the semantic coherence between the potential candidates and a predicate argument  $arg_n$ , we have exploited the selectional preferences in the same way as in previous SRL and implicit argument resolution works. First, we have designed a list of very general semantic categories. Second, we have semi-automatically assigned one of them to every predicate argument  $arg_n$  in PropBank and NomBank. For this, we have used the semantic annotation provided by the training documents of the CoNLL-2008 dataset. This annotation was performed automatically using the *SuperSense-Tagger* (Ciaramita and Altun, 2006) and includes

<sup>3</sup>Note that the algorithm could also consider sequences of closely related predicates.

named-entities and WordNet Super-Senses<sup>4</sup>. We have also defined a mapping between the semantic classes provided by the SuperSenseTagger and our seven semantic categories (see Table 1 for more details). Then, we have acquired the most common categories of each predicate argument  $arg_n$ . ImpAr algorithm also uses the *SuperSenseTagger* over the documents to be processed from BNB to check if the candidate belongs to the expected semantic category of the implicit argument to be filled.

Following the example above, [*Quest Medical Inc*] is tagged as an *ORGANIZATION* by the *SuperSenseTagger*. Therefore, it belongs to our semantic category *COGNITIVE*. As the semantic category for the implicit argument  $arg_0$  for the predicate **plan** has been recognized to be also *COGNITIVE*, [*Quest Medical Inc*] remains in the list of candidates as a possible filler.

Semantic category	Name-entities	Super-Senses
COGNITIVE	<i>PERSON</i>	noun.person
	<i>ORGANIZATION</i>	noun.group
	<i>ANIMAL</i>	noun.animal
	...	...
TANGIBLE	<i>PRODUCT</i>	noun.artifact
	<i>SUBSTANCE</i>	noun.object
	...	...
EVENTIVE	<i>GAME</i>	noun.act
	<i>DISEASE</i>	noun.communication
	...	...
RELATIVE	...	noun.shape
	...	noun.attribute
	...	...
LOCATIVE	<i>LOCATION</i>	noun.location
TIME	<i>DATE</i>	noun.time
	<i>QUANTITY</i>	noun.quantity
MESURABLE	<i>PERCENT</i>	...
	...	...

**Table 1:** Links between the semantic categories and some name-entities and super-senses.

**Saliency weighting.** In this process, the algorithm assigns to each candidate a set of saliency factors that scores its prominence. The *sentence recency* factor prioritizes the candidates that occur close to the same sentence of the predicate. The *subject*, *direct object*, *indirect object* and *non-adverbial* factors weight the saliency of the candidate depending on the syntactic role they belong to. Additionally, the head of these syntactic roles are prioritized by the *head* factor. We have used the same weights, listed in table 2, proposed by Lappin and Leass (1994).

In the example, candidate [*Quest Medical Inc*] is in the same sentence as the predicate **plan**, it

<sup>4</sup>Lexicographic files according to WordNet terminology.

Factor type	weight
Sentence recency	100
Subject	80
Direct object	50
Indirect object	40
Head	80
Non-adverbial	50

**Table 2:** Weights assigned to each saliency factor.

belongs to a subject, and, indeed, it is the head of that subject. Hence, the saliency score for this candidate is:  $100 + 80 + 80 = 260$ .

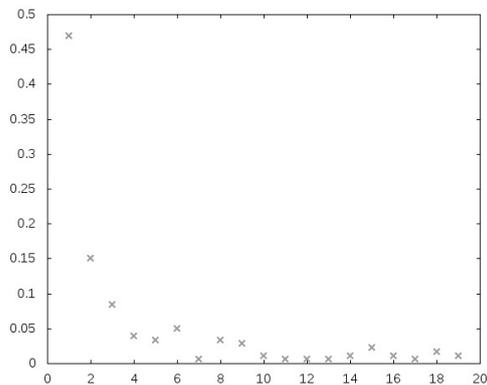
### 4.3 Damping the saliency of the default candidate

As the algorithm maintains the default candidate until an explicit filler appears, potential errors produced in the automatic selection process explained above can spread to distant implicit instances, specially when the saliency score of the default candidate is high. In order to reduce the impact of these errors we have included a damping factor that is applied sentence by sentence to the saliency value of the default candidate. ImpAr applies that damping factor,  $r$ , as follows. It assumes that, independently of the initial saliency assigned, 100 points of the saliency score came from the *sentence recency* factor. Then, the algorithm changes this value multiplying it by  $r$ . So, given a saliency score  $s$ , the value of the score in a following sentence,  $s'$ , is:

$$s' = s - 100 + 100 \cdot r$$

Obviously, the value of  $r$  must be defined without harming excessively those cases where the default candidate has been correctly identified. For this, we studied in the training dataset the cases of implicit arguments filled with the default candidate. Figure 1 shows that the influence of the default filler is much higher in near sentences that in more distance ones.

We tried to mimic a damping factor following this distribution. That is, to maintain high score saliency for the near sentences while strongly decreasing them in the subsequent ones. In this way, if the filler of the implicit argument is wrongly identified, the error only spreads to the nearest instances. If the identification is correct, a lower score for more distance sentences is not too harmful. The distribution shown in figure 1 follows an exponential decay, therefore we have described the damping factor as a curve like the following, where  $\alpha$  must be a value within 0 and 1:



**Figure 1:** Distances between the implicit argument and the default candidate. The  $y$  axis indicate the percentage of cases occurring in each sentence distance, expressed in  $x$

$$r = \alpha^d$$

In this function,  $d$  stands for the sentence distance and  $r$  for the damping factor to apply in that sentence. In this paper, we have decided to set the value of  $\alpha$  to 0.5.

$$r = 0.5^d$$

This value maintains the influence of the default fillers with high salience in near sentences. But it decreases that influence strongly in the following.

In order to illustrate the whole process we will use the previous example. In that case, *[Quest Medical Inc]* is selected as the  $arg_0$  of **plan** with a salience score of 260. Therefore *[Quest Medical Inc]* becomes the default  $arg_0$  of **plan**. In the following sentence the damping factor is:

$$0.5 = 0.5^1$$

Therefore, its salience score changes to  $260 - 100 + 100 \cdot 0.5 = 210$ . Then, the algorithm changes the default filler for  $arg_0$  only if it finds a candidate that scores higher in their current context. At two sentence distance, the resulting score for the default filler is  $260 - 100 + 100 \cdot 0.25 = 185$ . In this way, at more distance sentences, the influence of the default filler of  $arg_0$  becomes smaller.

## 5 Evaluation

In order to evaluate the performance of the ImpAr algorithm, we have followed the evaluation method presented by Gerber and Chai (2010, 2012). For every argument position in the gold-standard the scorer expects a single predicted constituent to fill in. In order to evaluate the correct span of a constituent, a prediction is scored using the Dice coefficient:

$$\frac{2|Predicted \cap True|}{|Predicted| + |True|}$$

The function above relates the set of tokens that form a predicted constituent, *Predicted*, and the set of tokens that are part of an annotated constituent in the gold-standard, *True*. For each missing argument, the gold-standard includes the whole coreference chain of the filler. Therefore, the scorer selects from all coreferent mentions the highest Dice value. If the predicted span does not cover the head of the annotated filler, the scorer returns zero. Then, *Precision* is calculated by the sum of all prediction scores divided by the number of attempts carried out by the system. *Recall* is equal to the sum of the prediction scores divided by the number of actual annotations in the gold-standard. F-measure is calculated as the harmonic mean of recall and precision.

Traditionally, there have been two approaches to develop SRL systems, one based on constituent trees and the other one based on syntactic dependencies. Additionally, the evaluation of both types of systems has been performed differently. For constituent based SRL systems the scorers evaluate the correct span of the filler, while for dependency based systems the scorer just check if the systems are able to capture the head token of the filler. As shown above, previous works in implicit argument resolution proposed a metric that involves the correct identification of the whole span of the filler. ImpAr algorithm works with syntactic dependencies and therefore it only returns the head token of the filler. In order to compare our results with previous works, we had to apply some simple heuristics to guess the correct span of the filler. Obviously, this process inserts some noise in the final evaluation.

We have performed a first evaluation over the test set used in (Gerber and Chai, 2010). This dataset contains 437 predicate instances but just 246 argument positions are implicitly filled. Table 3 includes the results obtained by ImpAr, the results of the system presented by Gerber and Chai (2010) and the baseline proposed for the task. Best results are marked in bold<sup>5</sup>. For all predicates, ImpAr improves over the baseline (19.3 points higher in the overall  $F_1$ ). Our system also outperforms the one presented by Gerber and Chai (2010). Interestingly, both systems present very different performances predicate by predicate. For

<sup>5</sup>No proper significance test can be carried out without the the full predictions of all systems involved.

			Baseline	Gerber & Chai			ImpAr		
	#Inst.	#Imp.	$F_1$	$P$	$R$	$F_1$	$P$	$R$	$F_1$
sale	64	65	36.2	<b>47.2</b>	<b>41.7</b>	<b>44.2</b>	41.2	39.4	40.3
price	121	53	15.4	36.0	32.6	34.2	<b>53.3</b>	<b>53.3</b>	<b>53.3</b>
investor	78	35	9.8	36.8	<b>40.0</b>	38.4	<b>43.0</b>	39.5	<b>41.2</b>
bid	19	26	32.3	23.8	19.2	21.3	<b>52.9</b>	<b>51.0</b>	<b>52.0</b>
plan	25	20	38.5	<b>78.6</b>	<b>55.0</b>	<b>64.7</b>	40.7	40.7	40.7
cost	25	17	34.8	<b>61.1</b>	<b>64.7</b>	<b>62.9</b>	56.1	50.2	53.0
loss	30	12	52.6	<b>83.3</b>	<b>83.3</b>	<b>83.3</b>	68.4	63.5	65.8
loan	11	9	18.2	<b>42.9</b>	<b>33.3</b>	<b>37.5</b>	25.0	20.0	22.2
investment	21	8	0.0	40.0	25.0	30.8	<b>47.6</b>	<b>35.7</b>	<b>40.8</b>
fund	43	6	0.0	14.3	16.7	15.4	<b>66.7</b>	<b>33.3</b>	<b>44.4</b>
Overall	437	246	26.5	44.5	40.4	42.3	<b>47.9</b>	<b>43.8</b>	<b>45.8</b>

**Table 3:** Evaluation with the test. The results from (Gerber and Chai, 2010) are included.

			Baseline	Gerber & Chai			ImpAr		
	#Inst.	#Imp.	$F_1$	$P$	$R$	$F_1$	$P$	$R$	$F_1$
sale	184	181	37.3	<b>59.2</b>	<b>44.8</b>	<b>51.0</b>	44.3	43.3	43.8
price	216	138	34.6	<b>56.0</b>	48.7	52.1	55.0	<b>54.5</b>	<b>54.7</b>
investor	160	108	5.1	<b>46.7</b>	39.8	<b>43.0</b>	28.2	27.0	27.6
bid	88	124	23.8	60.0	36.3	<b>45.2</b>	48.4	<b>41.8</b>	45.0
plan	100	77	32.3	<b>59.6</b>	44.1	<b>50.7</b>	47.0	<b>47.0</b>	47.0
cost	101	86	17.8	<b>62.5</b>	<b>50.9</b>	<b>56.1</b>	49.2	43.7	46.2
loss	104	62	54.7	<b>72.5</b>	<b>59.7</b>	<b>65.5</b>	63.0	58.2	60.5
loan	84	82	31.2	<b>67.2</b>	<b>50.0</b>	<b>57.3</b>	56.4	45.6	50.6
investment	102	52	15.5	32.9	<b>34.2</b>	33.6	<b>41.2</b>	30.9	<b>35.4</b>
fund	108	56	15.5	<b>80.0</b>	35.7	49.4	55.6	<b>44.6</b>	<b>49.5</b>
Overall	1,247	966	28.9	<b>57.9</b>	<b>44.5</b>	<b>50.3</b>	47.7	43.0	45.3

**Table 4:** Evaluation with the full dataset. The results from (Gerber and Chai, 2012) are included.

instance, our system obtains much higher results for the predicates **bid** and **fund**, while much lower for **loss** and **loan**. In general, ImpAr seems to be more robust since it obtains similar performances for all predicates. In fact, the standard deviation,  $\sigma$ , of  $F_1$  measure is 10.98 for ImpAr while this value for the (Gerber and Chai, 2010) system is 20.00.

In a more recent work, Gerber and Chai (2012) presented some improvements of their previous results. In this work, they extended the evaluation of their model using the whole dataset and not just the testing documents. Applying a cross-validated approach they tried to solve some problems that they found in the previous evaluation, like the small size of the testing set. For this work, they also studied a wider set of features, specially, they experimented with some statistics learnt from parts of GigaWord automatically annotated. Table 4 shows that the improvement over their previous system was remarkable. The system also seems to be more stable across predicates. For comparison purposes, we also included the performance of ImpAr applied over the whole dataset.

The results in table 4 show that, although ImpAr still achieves the best results in some cases, this time, it cannot beat the overall results obtained by

the supervised model. In fact, both systems obtain a very similar recall, but the system from (Gerber and Chai, 2012) obtains much higher precision. In both cases, the  $\sigma$  value of  $F_1$  is reduced, 8.81 for ImpAr and 8.21 for (Gerber and Chai, 2012). However, ImpAr obtains very similar performance independently of the testing dataset what proves the robustness of the algorithm. This suggests that our algorithm can obtain strong results also for other corpus and predicates. Instead, the supervised approach would need a large amount of manual annotations for every predicate to be processed.

## 6 Discussion

### 6.1 Component Analysis

In order to assess the contribution of each system component, we also tested the performance of ImpAr algorithm when disabling only one of its components. With this evaluations we pretend to sight the particular contribution of each component. In table 5 we present the results obtained in the following experiments for the two testing sets explained in section 5:

- Exp1: The damping factor is disabled. All selected fillers maintain the same salience over

all sentences.

- Exp2: Only explicit fillers are considered as candidates<sup>6</sup>.
- Exp3: No default fillers are considered as candidates.

As expected, we observe a very similar performances in both datasets. Additionally, the highest loss appears when the default fillers are ruled out (Exp3). In particular, it also seems that the explicit information from previous predicates provides the most correct evidence (Exp2). Also note that for Exp2, the system obtains the highest precision. This means that the most accurate cases are obtained by previous explicit antecedents.

	test			full		
	<i>P</i>	<i>R</i>	<i>F</i> <sub>1</sub>	<i>P</i>	<i>R</i>	<i>F</i> <sub>1</sub>
full	47.9	<b>43.8</b>	<b>45.8</b>	47.7	<b>43.0</b>	<b>45.3</b>
Exp1	45.7	41.8	43.6	47.1	42.5	44.8
Exp2	<b>51.2</b>	24.6	33.2	<b>55.3</b>	25.5	34.9
Exp3	34.6	29.7	31.9	34.8	28.9	31.5
Exp4	<b>42.6</b>	<b>37.9</b>	<b>40.1</b>	<b>37.5</b>	<b>31.2</b>	<b>34.1</b>
Exp5	38.8	34.5	36.5	35.7	29.7	32.4
Exp6	53.3	48.7	50.9	52.4	47.2	49.6

**Table 5:** Exp1, Exp2 and Exp3 correspond to ablations of the components. Exp3 and Exp4 are experiments over the cases that are not solved by explicit antecedents. Exp6 evaluates the system capturing just the head tokens of the constituents.

As Exp1 also includes instances with explicit antecedents, and for these cases the damping factor component has no effect, we have designed two additional experiments:

- Exp4: Full system for the cases not solved by explicit antecedents.
- Exp5: As in Exp4 but with the damping factor disabled.

As expected, now the contribution of the dumping factor seems to be more relevant, in particular, for the *test* dataset.

## 6.2 Correct span of the fillers

As explained in Section 5, our algorithm works with syntactic dependencies and its predictions only return the head token of the filler. Obtaining the correct constituents from syntactic dependencies is not trivial. In this work we have applied a simple heuristic that returns all the descendant

<sup>6</sup>That is, implicit arguments without explicit antecedents are not filled.

tokens of the predicted head token. This naive process inserts some noise to the evaluation of the system. For example, from the following sentence our system gives the following prediction for an implicit  $arg_1$  of an instance of the predicate **sale**:

Ports of Call Inc. reached agreements to sell its remaining seven aircraft [ $arg_1$  to buyers] that weren't disclosed.

But the actual gold-standard annotation is: [ $arg_1$  buyers that weren't disclosed]. Although the head of the constituent, *buyers*, is correctly captured by ImpAr, the final prediction is heavily penalized by the scoring method. Table 5 presents the results of ImpAr when evaluating the head tokens of the constituents only (Exp6). These results show that the current performance of our system can be easily improved applying a more accurate process for capturing the correct span.

## 7 Conclusions and Future Work

In this work we have presented a robust deterministic approach for implicit Semantic Role Labeling. The method exploits a very simple but relevant discursive coherence property that holds over explicit and implicit arguments of closely related nominal and verbal predicates. This property states that if several instances of the same predicate appear in a well-written discourse, it is very likely that they maintain the same argument fillers. We have shown the importance of this phenomenon for recovering the implicit information about semantic roles. To our knowledge, this is the first empirical study that proves this phenomenon.

Based on these observations, we have developed a new deterministic algorithm, ImpAr, that obtains very competitive and robust performances with respect to supervised approaches. That is, it can be applied where there is no available manual annotations to train. The code of this algorithm is publicly available and can be applied to any document. As input it only needs the document with explicit semantic role labeling and Super-Sense annotations. These annotations can be easily obtained from plain text using available tools<sup>7</sup>, what makes this algorithm the first effective tool available for implicit SRL.

As it can be easily seen, ImpAr has a large margin for improvement. For instance, providing more accurate spans for the fillers. We also plan

<sup>7</sup>We recommend mate-tools (Björkelund et al., 2009) and SuperSenseTagger (Ciaramita and Altun, 2006).

to test alternative approaches to solve the arguments without explicit antecedents. For instance, our system can also profit from additional annotations like coreference, that has proved its utility in previous works. Finally, we also plan to study our approach on different languages and datasets (for instance, the SemEval-2010 dataset).

## 8 Acknowledgment

We are grateful to the anonymous reviewers for their insightful comments. This work has been partially funded by SKaTer (TIN2012-38584-C06-02), OpeNER (FP7-ICT-2011-SME-DCL-296451) and NewsReader (FP7-ICT-2011-8-316404), as well as the READERS project with the financial support of MINECO, ANR (convention ANR-12-CHRI-0004-03) and EPSRC (EP/K017845/1) in the framework of ERA-NET CHIST-ERA (UE FP7/2007-2013).

## References

- Baker, C. F., C. J. Fillmore, and J. B. Lowe (1998). The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, ACL '98, Montreal, Quebec, Canada, pp. 86–90.
- Björkelund, A., L. Hafdell, and P. Nugues (2009). Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, Boulder, Colorado, USA, pp. 43–48.
- Carreras, X. and L. Màrquez (2005). Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning*, CoNLL '05, Ann Arbor, Michigan, USA, pp. 152–164.
- Chen, D., N. Schneider, D. Das, and N. A. Smith (2010). Semafor: Frame argument resolution with log-linear models. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, Los Angeles, California, USA, pp. 264–267.
- Ciaramita, M. and Y. Altun (2006). Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, Sydney, Australia, pp. 594–602.
- Dahl, D. A., M. S. Palmer, and R. J. Passonneau (1987). Nominalizations in pundit. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, ACL '87, Stanford, California, USA, pp. 131–139.
- Fellbaum, C. (1998). *WordNet: an electronic lexical database*. MIT Press.
- Gerber, M. and J. Chai (2012, December). Semantic role labeling of implicit arguments for nominal predicates. *Computational Linguistics* 38(4), 755–798.
- Gerber, M. and J. Y. Chai (2010). Beyond nonbank: a study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, Uppsala, Sweden, pp. 1583–1592.
- Gildea, D. and D. Jurafsky (2000). Automatic labeling of semantic roles. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, Hong Kong, pp. 512–520.
- Gildea, D. and D. Jurafsky (2002, September). Automatic labeling of semantic roles. *Computational Linguistics* 28(3), 245–288.
- Gorinski, P., J. Ruppenhofer, and C. Sporleder (2013). Towards weakly supervised resolution of null instantiations. In *Proceedings of the 10th International Conference on Computational Semantics*, IWCS '13, Potsdam, Germany, pp. 119–130.
- Hajič, J., M. Ciaramita, R. Johansson, D. Kawahara, M. A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štěpánek, P. Straňák, M. Surdeanu, N. Xue, and Y. Zhang (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, Boulder, Colorado, USA, pp. 1–18.
- Laparra, E. and G. Rigau (2012). Exploiting explicit annotations and semantic types for implicit argument resolution. In *6th IEEE International Conference on Semantic Computing*, ICSC '12, Palermo, Italy, pp. 75–78.

- Laparra, E. and G. Rigau (2013). Sources of evidence for implicit argument resolution. In *Proceedings of the 10th International Conference on Computational Semantics, IWCS '13*, Potsdam, Germany, pp. 155–166.
- Lappin, S. and H. J. Leass (1994, December). An algorithm for pronominal anaphora resolution. *Computational Linguistics* 20(4), 535–561.
- Meyers, A., R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman (2004). The nombank project: An interim report. In *Proceedings of the NAACL/HLT Workshop on Frontiers in Corpus Annotation, HLT-NAACL '04*, Boston, Massachusetts, USA, pp. 24–31.
- Moor, T., M. Roth, and A. Frank (2013). Predicate-specific annotations for implicit role binding: Corpus annotation, data analysis and evaluation experiments. In *Proceedings of the 10th International Conference on Computational Semantics, IWCS '13*, Potsdam, Germany, pp. 369–375.
- Palmer, M., D. Gildea, and P. Kingsbury (2005, March). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics* 31(1), 71–106.
- Palmer, M. S., D. A. Dahl, R. J. Schiffman, L. Hirschman, M. Linebarger, and J. Dowding (1986). Recovering implicit information. In *Proceedings of the 24th annual meeting on Association for Computational Linguistics, ACL '86*, New York, New York, USA, pp. 10–19.
- Raghunathan, K., H. Lee, S. Rangarajan, N. Chambers, M. Surdeanu, D. Jurafsky, and C. Manning (2010). A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, Cambridge, Massachusetts, USA, pp. 492–501.
- Ruppenhofer, J., P. Gorinski, and C. Sporleder (2011). In search of missing arguments: A linguistic approach. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011, RANLP '11*, Hissar, Bulgaria, pp. 331–338.
- Ruppenhofer, J., C. Sporleder, R. Morante, C. Baker, and M. Palmer (2010). Semeval-2010 task 10: Linking events and their participants in discourse. In *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10*, Los Angeles, California, USA, pp. 45–50.
- Silberer, C. and A. Frank (2012). Casting implicit role linking as an anaphora resolution task. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics, \*SEM '12*, Montréal, Canada, pp. 1–10.
- Surdeanu, M., R. Johansson, A. Meyers, L. Màrquez, and J. Nivre (2008). The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Natural Language Learning, CoNLL '08*, Manchester, United Kingdom, pp. 159–177.
- Tetreault, J. R. (2002). Implicit role reference. In *International Symposium on Reference Resolution for Natural Language Processing*, Alicante, Spain, pp. 109–115.
- Tonelli, S. and R. Delmonte (2010). Venses++: Adapting a deep semantic processing system to the identification of null instantiations. In *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10*, Los Angeles, California, USA, pp. 296–299.
- Tonelli, S. and R. Delmonte (2011). Desperately seeking implicit arguments in text. In *Proceedings of the ACL 2011 Workshop on Relational Models of Semantics, RELMS '11*, Portland, Oregon, USA, pp. 54–62.
- Whittemore, G., M. Macpherson, and G. Carlson (1991). Event-building through role-filling and anaphora resolution. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics, ACL '91*, Berkeley, California, USA, pp. 17–24.

# Cross-lingual Transfer of Semantic Role Labeling Models

Mikhail Kozhevnikov and Ivan Titov

Saarland University, Postfach 15 11 50

66041 Saarbrücken, Germany

{mkozhevn|titov}@mmci.uni-saarland.de

## Abstract

Semantic Role Labeling (SRL) has become one of the standard tasks of natural language processing and proven useful as a source of information for a number of other applications. We address the problem of transferring an SRL model from one language to another using a shared feature representation. This approach is then evaluated on three language pairs, demonstrating competitive performance as compared to a state-of-the-art unsupervised SRL system and a cross-lingual annotation projection baseline. We also consider the contribution of different aspects of the feature representation to the performance of the model and discuss practical applicability of this method.

## 1 Background and Motivation

Semantic role labeling has proven useful in many natural language processing tasks, such as question answering (Shen and Lapata, 2007; Kaisser and Webber, 2007), textual entailment (Sammons et al., 2009), machine translation (Wu and Fung, 2009; Liu and Gildea, 2010; Gao and Vogel, 2011) and dialogue systems (Basili et al., 2009; van der Plas et al., 2009).

Multiple models have been designed to automatically predict semantic roles, and a considerable amount of data has been annotated to train these models, if only for a few more popular languages. As the annotation is costly, one would like to leverage existing resources to minimize the human effort required to construct a model for a new language.

A number of approaches to the construction of semantic role labeling models for new languages

have been proposed. On one end of the scale is unsupervised SRL, such as Grenager and Manning (2006), which requires some expert knowledge, but no labeled data. It clusters together arguments that should bear the same semantic role, but does not assign a particular role to each cluster. On the other end is annotating a new dataset from scratch. There are also intermediate options, which often make use of similarities between languages. This way, if an accurate model exists for one language, it should help simplify the construction of a model for another, related language.

The approaches in this third group often use parallel data to bridge the gap between languages. Cross-lingual annotation projection systems (Padó and Lapata, 2009), for example, propagate information directly via word alignment links. However, they are very sensitive to the quality of parallel data, as well as the accuracy of a source-language model on it.

An alternative approach, known as cross-lingual model transfer, or cross-lingual model adaptation, consists of modifying a source-language model to make it directly applicable to a new language. This usually involves constructing a shared feature representation across the two languages. McDonald et al. (2011) successfully apply this idea to the transfer of dependency parsers, using part-of-speech tags as the shared representation of words. A later extension of Täckström et al. (2012) enriches this representation with cross-lingual word clusters, considerably improving the performance.

In the case of SRL, a shared representation that is purely syntactic is likely to be insufficient, since structures with different semantics may be realized by the same syntactic construct, for example “in August” vs “in Britain”. However with the help of recently introduced cross-lingual word represen-

tations, such as the cross-lingual clustering mentioned above or cross-lingual distributed word representations of Klementiev et al. (2012), we may be able to transfer models of shallow semantics in a similar fashion.

In this work we construct a shared feature representation for a pair of languages, employing cross-lingual representations of syntactic and lexical information, train a semantic role labeling model on one language and apply it to the other one. This approach yields an SRL model for a new language at a very low cost, effectively requiring only a source language model and parallel data.

We evaluate on five (directed) language pairs – EN-ZH, ZH-EN, EN-CZ, CZ-EN and EN-FR, where EN, FR, CZ and ZH denote English, French, Czech and Chinese, respectively. The transferred model is compared against two baselines: an unsupervised SRL system and a model trained on the output of a cross-lingual annotation projection system.

In the next section we will describe our setup, then in section 3 present the shared feature representation we use, discuss the evaluation data and other technical aspects in section 4, present the results and conclude with an overview of related work.

## 2 Setup

The purpose of the study is not to develop a yet another semantic role labeling system – any existing SRL system can (after some modification) be used in this setup – but to assess the practical applicability of cross-lingual model transfer to this problem, compare it against the alternatives and identify its strong/weak points depending on a particular setup.

### 2.1 Semantic Role Labeling Model

We consider the dependency-based version of semantic role labeling as described in Hajič et al. (2009) and transfer an SRL model from one language to another. We only consider verbal predicates and ignore the predicate disambiguation stage. We also assume that the predicate identification information is available – in most languages it can be obtained using a relatively simple heuristic based on part-of-speech tags.

The model performs argument identification and classification (Johansson and Nugues, 2008) separately in a pipeline – first each candidate is

classified as being or not being a head of an argument phrase with respect to the predicate in question and then each of the arguments is assigned a role from a given inventory. The model is factorized over arguments – the decisions regarding the classification of different arguments are made independently of each other.

With respect to the use of syntactic annotation we consider two options: using an existing dependency parser for the target language and obtaining one by means of cross-lingual transfer (see section 4.2).

Following McDonald et al. (2011), we assume that a part-of-speech tagger is available for the target language.

### 2.2 SRL in the Low-resource Setting

Several approaches have been proposed to obtain an SRL model for a new language with little or no manual annotation. Unsupervised SRL models (Lang and Lapata, 2010) cluster the arguments of predicates in a given corpus according to their semantic roles. The performance of such models can be impressive, especially for those languages where semantic roles correlate strongly with syntactic relation of the argument to its predicate. However, assigning meaningful role labels to the resulting clusters requires additional effort and the model’s parameters generally need some adjustment for every language.

If the necessary resources are already available for a closely related language, they can be utilized to facilitate the construction of a model for the target language. This can be achieved either by means of cross-lingual annotation projection (Yarowsky et al., 2001) or by cross-lingual model transfer (Zeman and Resnik, 2008).

This last approach is the one we are considering in this work, and the other two options are treated as baselines. The unsupervised model will be further referred to as UNSUP and the projection baseline as PROJ.

### 2.3 Evaluation Measures

We use the  $F_1$  measure as a metric for the argument identification stage and accuracy as an aggregate measure of argument classification performance. When comparing to the unsupervised SRL system the clustering evaluation measures are used instead. These are purity and collocation

$$PU = \frac{1}{N} \sum_i \max_j |G_j \cap C_i|$$

$$CO = \frac{1}{N} \sum_j \max_i |G_j \cap C_i|,$$

where  $C_i$  is the set of arguments in the  $i$ -th induced cluster,  $G_j$  is the set of arguments in the  $j$ th gold cluster and  $N$  is the total number of arguments. We report the harmonic mean of the two (Lang and Lapata, 2011) and denote it  $F_1^c$  to avoid confusing it with the supervised metric.

### 3 Model Transfer

The idea of this work is to abstract the model away from the particular source language and apply it to a new one. This setup requires that we use the same feature representation for both languages, for example part-of-speech tags and dependency relation labels should be from the same inventory.

Some features are not applicable to certain languages because the corresponding phenomena are absent in them. For example, consider a strongly inflected language and an analytic one. While the latter can usually convey the information encoded in the word form in the former one (number, gender, etc.), finding a shared feature representation for such information is non-trivial. In this study we will confine ourselves to those features that are applicable to all languages in question, namely: part-of-speech tags, syntactic dependency structures and representations of the word’s identity.

#### 3.1 Lexical Information

We train a model on one language and apply it to a different one. In order for this to work, the words of the two languages have to be mapped into a common feature space. It is also desirable that closely related words from both languages have similar representations in this space.

*Word mapping.* The first option is simply to use the source language words as the shared representation. Here every source language word would have itself as its representation and every target word would map into a source word that corresponds to it. In other words, we supply the model with a gloss of the target sentence.

The mapping (bilingual dictionary) we use is derived from a word-aligned parallel corpus, by identifying, for each word in the target language,

the word in the source language it is most often aligned to.

*Cross-lingual clusters.* There is no guarantee that each of the words in the evaluation data is present in our dictionary, nor that the corresponding source-language word is present in the training data, so the model would benefit from the ability to generalize over closely related words. This can, for example, be achieved by using cross-lingual word clusters induced in Täckström et al. (2012). We incorporate these clusters as features into our model.

#### 3.2 Syntactic Information

*Part-of-speech Tags.* We map part-of-speech tags into the universal tagset following Petrov et al. (2012). This may have a negative effect on the performance of a monolingual model, since most part-of-speech tagsets are more fine-grained than the universal POS tags considered here. For example Penn Treebank inventory contains 36 tags and the universal POS tagset – only 12. Since the finer-grained POS tags often reflect more language-specific phenomena, however, they would only be useful for very closely related languages in the cross-lingual setting.

The universal part-of-speech tags used in evaluation are derived from gold-standard annotation for all languages except French, where predicted ones had to be used instead.

*Dependency Structure.* Another important aspect of syntactic information is the dependency structure. Most dependency relation inventories are language-specific, and finding a shared representation for them is a challenging problem. One could map dependency relations into a simplified form that would be shared between languages, as it is done for part-of-speech tags in Petrov et al. (2012). The extent to which this would be useful, however, depends on the similarity of syntactic-semantic interfaces of the languages in question.

In this work we discard the dependency relation labels where the inventories do not match and only consider the unlabeled syntactic dependency graph. Some discrepancies, such as variations in attachment order, may be present even there, but this does not appear to be the case with the datasets we use for evaluation. If a target language is poor in resources, one can obtain a dependency parser for the target language by means of cross-lingual model transfer (Zeman and Resnik, 2008). We

take this into account and evaluate both using the original dependency structures and the ones obtained by means of cross-lingual model transfer.

### 3.3 The Model

The model we use is based on that of Björkelund et al. (2009). It is comprised of a set of linear classifiers trained using Liblinear (Fan et al., 2008). The feature model was modified to accommodate the cross-lingual cluster features and the reranker component was not used.

We do not model the interaction between different argument roles in the same predicate. While this has been found useful, in the cross-lingual setup one has to be careful with the assumptions made. For example, modeling the sequence of roles using a Markov chain (Thompson et al., 2003) may not work well in the present setting, especially between distant languages, as the order or arguments is not necessarily preserved. Most constraints that prove useful for SRL (Chang et al., 2007) also require customization when applied to a new language, and some rely on language-specific resources, such as a valency lexicon. Taking into account the interaction between different arguments of a predicate is likely to improve the performance of the transferred model, but this is outside the scope of this work.

### 3.4 Feature Selection

Compatibility of feature representations is necessary but not sufficient for successful model transfer. We have to make sure that the features we use are predictive of similar outcomes in the two languages as well.

Depending on the pair of languages in question, different aspects of the feature representation will retain or lose their predictive power. We can be reasonably certain that the identity of an argument word is predictive of its semantic role in any language, but it might or might not be true of, for example, the word directly preceding the argument word. It is therefore important to pre-

POS	part-of-speech tags
Synt	unlabeled dependency graph
Cls	cross-lingual word clusters
Gloss	glossed word forms
Deprel	dependency relations

Table 1: Feature groups.

vent the model from capturing overly specific aspects of the source language, which we do by confining the model to first-order features. We also avoid feature selection, which, performed on the source language, is unlikely to help the model to better generalize to the target one. The experiments confirm that feature selection and the use of second-order features degrade the performance of the transferred model.

### 3.5 Feature Groups

For each word, we use its part-of-speech tag, cross-lingual cluster id, word identity (glossed, when evaluating on the target language) and its dependency relation to its parent. Features associated with an argument word include the attributes of the predicate word, the argument word, its parent, siblings and children, and the words directly preceding and following it. Also included are the sequences of part-of-speech tags and dependency relations on the path between the predicate and the argument.

Since we are also interested in the impact of different aspects of the feature representation, we divide the features into groups as summarized in table 1 and evaluate their respective contributions to the performance of the model. If a feature group is enabled – the model has access to the corresponding source of information. For example, if only POS group is enabled, the model relies on the part-of-speech tags of the argument, the predicate and the words to the right and left of the argument word. If Synt is enabled too, it also uses the POS tags of the argument’s parent, children and siblings.

Word order information constitutes an implicit group that is always available. It includes the Position feature, which indicates whether the argument is located to the left or to the right of the predicate, and allows the model to look up the attributes of the words directly preceding and following the argument word. The model we compare against the baselines uses all applicable feature groups (Deprel is only used in EN-CZ and CZ-EN experiments with original syntax).

## 4 Evaluation

### 4.1 Datasets and Preprocessing

Evaluation of the cross-lingual model transfer requires a rather specific kind of dataset. Namely, the data in both languages has to be annotated

with the same set of semantic roles following the same (or compatible) guidelines, which is seldom the case. We have identified three language pairs for which such resources are available: English-Chinese, English-Czech and English-French.

The evaluation datasets for English and Chinese are those from the CoNLL Shared Task 2009 (Hajič et al., 2009) (henceforth CoNLL-ST). Their annotation in the CoNLL-ST is not identical, but the guidelines for “core” semantic roles are similar (Kingsbury et al., 2004), so we evaluate only on core roles here. The data for the second language pair is drawn from the Prague Czech-English Dependency Treebank 2.0 (Hajič et al., 2012), which we converted to a format similar to that of CoNLL-ST<sup>1</sup>. The original annotation uses the tectogrammatical representation (Hajič, 2002) and an inventory of semantic roles (or *functions*), most of which are interpretable across various predicates. Also note that the syntactic annotation of English and Czech in PCEDT 2.0 is quite similar (to the extent permitted by the difference in the structure of the two languages) and we can use the dependency relations in our experiments.

For English-French, the English CoNLL-ST dataset was used as a source and the model was evaluated on the manually annotated dataset from van der Plas et al. (2011). The latter contains one thousand sentences from the French part of the Europarl (Koehn, 2005) corpus, annotated with semantic roles following an adapted version of PropBank (Palmer et al., 2005) guidelines. The authors perform annotation projection from English to French, using a joint model of syntax and semantics and employing heuristics for filtering. We use a model trained on the output of this projection system as one of the baselines. The evaluation dataset is relatively small in this case, so we perform the transfer only one-way, from English to French.

The part-of-speech tags in all datasets were replaced with the universal POS tags of Petrov et al. (2012). For Czech, we have augmented the mappings to account for the tags that were not present in the datasets from which the original mappings were derived. Namely, tag “t” is mapped to “VERB” and “Y” – to “PRON”.

We use parallel data to construct a bilingual dictionary used in word mapping, as well as in the projection baseline. For English-Czech

and English-French, the data is drawn from Europarl (Koehn, 2005), for English-Chinese – from MultiUN (Eisele and Chen, 2010). The word alignments were obtained using GIZA++ (Och and Ney, 2003) and the intersection heuristic.

## 4.2 Syntactic Transfer

In the low-resource setting, we cannot always rely on the availability of an accurate dependency parser for the target language. If one is not available, the natural solution would be to use cross-lingual model transfer to obtain it.

Unfortunately, the models presented in the previous work, such as Zeman and Resnik (2008), McDonald et al. (2011) and Täckström et al. (2012), were not made available, so we reproduced the direct transfer algorithm of McDonald et al. (2011), using Malt parser (Nivre, 2008) and the same set of features. We did not reimplement the projected transfer algorithm, however, and used the default training procedure instead of perceptron-based learning. The dependency structure thus obtained is, of course, only a rough approximation – even a much more sophisticated algorithm may not perform well when transferring syntax between such languages as Czech and English, given the inherent difference in their structure. The scores are shown in table 2.

We will henceforth refer to the syntactic annotations that were provided with the datasets as *original*, as opposed to the annotations obtained by means of syntactic transfer.

## 4.3 Baselines

*Unsupervised Baseline:* We are using a version of the unsupervised semantic role induction system of Titov and Klementiev (2012a) adapted to

Setup	UAS, %
EN-ZH	35
ZH-EN	42
EN-CZ	36
CZ-EN	39
EN-FR	67

Table 2: Syntactic transfer accuracy, unlabeled attachment score (percent). Note that in case of French we evaluate against the output of a supervised system, since manual annotation is not available for this dataset. This score does not reflect the true performance of syntactic transfer.

<sup>1</sup>see <http://www.ml4nlp.de/code-and-data/treeex2conll>

the shared feature representation considered in order to make the scores comparable with those of the transfer model and, more importantly, to enable evaluation on transferred syntax. Note that the original system, tailored to a more expressive language-specific syntactic representation and equipped with heuristics to identify active/passive voice and other phenomena, achieves higher scores than those we report here.

*Projection Baseline:* The projection baseline we use for English-Czech and English-Chinese is a straightforward one: we label the source side of a parallel corpus using the source-language model, then identify those verbs on the target side that are aligned to a predicate, mark them as predicates and propagate the argument roles in the same fashion. A model is then trained on the resulting training data and applied to the test set.

For English-French we instead use the output of a fully featured projection model of van der Plas et al. (2011), published in the CLASSiC project.

## 5 Results

In order to ensure that the results are consistent, the test sets, except for the French one, were partitioned into five equal parts (of 5 to 10 thousand sentences each, depending on the dataset) and the evaluation performed separately on each one. All evaluation figures for English, Czech or Chinese below are the average values over the five subsets. In case of French, the evaluation dataset is too small to split it further, so instead we ran the evaluation five times on a randomly selected 80% sample of the evaluation data and averaged over those. In both cases the results are consistent over the subsets, the standard deviation does not exceed 0.5% for the transfer system and projection baseline and 1% for the unsupervised system.

### 5.1 Argument Identification

We summarize the results in table 3. Argument identification is known to rely heavily on syntactic information, so it is unsurprising that it proves inaccurate when transferred syntax is used. Our simple projection baseline suffers from the same problem. Even with original syntactic information available, the performance of argument identification is moderate. Note that the model of (van der Plas et al., 2011), though relying on more expressive syntax, only outperforms the transferred system by 3% ( $F_1$ ) on this task.

Setup	Syntax	TRANS	PROJ
EN-ZH	trans	34.5	13.9
ZH-EN	trans	32.6	15.6
EN-CZ	trans	46.3	12.4
CZ-EN	trans	42.3	22.2
EN-FR	trans	61.6	43.5
EN-ZH	orig	51.7	19.6
ZH-EN	orig	53.2	29.7
EN-CZ	orig	63.9	59.3
CZ-EN	orig	67.3	60.9
EN-FR	orig	71.0	51.3

Table 3: Argument identification, transferred model vs. projection baseline,  $F_1$ .

Most unsupervised SRL approaches assume that the argument identification is performed by some external means, for example heuristically (Lang and Lapata, 2011). Such heuristics or unsupervised approaches to argument identification (Abend et al., 2009) can also be used in the present setup.

### 5.2 Argument Classification

In the following tables, TRANS column contains the results for the transferred system, UNSUP – for the unsupervised baseline and PROJ – for projection baseline. We highlight in bold the higher score where the difference exceeds twice the maximum of the standard deviation estimates of the two results.

Table 4 presents the unsupervised evaluation results. Note that the unsupervised model performs as well as the transferred one or better where the

Setup	Syntax	TRANS	UNSUP
EN-ZH	trans	<b>83.3</b>	73.9
ZH-EN	trans	<b>79.2</b>	67.6
EN-CZ	trans	66.4	66.1
CZ-EN	trans	68.2	68.7
EN-FR	trans	<b>74.6</b>	65.1
EN-ZH	orig	84.5	<b>89.7</b>
ZH-EN	orig	79.2	<b>83.0</b>
EN-CZ	orig	74.1	74.0
CZ-EN	orig	74.6	<b>76.7</b>
EN-FR	orig	73.3	72.3

Table 4: Argument classification, transferred model vs. unsupervised baseline in terms of the clustering metric  $F_1^c$  (see section 2.3).

Setup	Syntax	TRANS	PROJ
EN-ZH	trans	<b>70.1</b>	69.2
ZH-EN	trans	<b>65.6</b>	61.3
EN-CZ	trans	<b>50.1</b>	46.3
CZ-EN	trans	53.3	<b>54.7</b>
EN-FR	trans	65.1	<b>66.1</b>
EN-ZH	orig	<b>71.7</b>	69.7
ZH-EN	orig	<b>66.1</b>	64.4
EN-CZ	orig	<b>59.0</b>	53.2
CZ-EN	orig	61.0	60.8
EN-FR	orig	63.0	<b>68.0</b>

Table 5: Argument classification, transferred model vs. projection baseline, accuracy.

original syntactic dependencies are available. In the more realistic scenario with transferred syntax, however, the transferred model proves more accurate.

In table 5 we compare the transferred system with the projection baseline. It is easy to see that the scores vary strongly depending on the language pair, due to both the difference in the annotation scheme used and the degree of relatedness between the languages. The drop in performance when transferring the model to another language is large in every case, though, see table 6.

Setup	Target	Source
EN-ZH	71.7	87.1
ZH-EN	66.1	86.2
EN-CZ	59.0	80.1
CZ-EN	61.0	75.4
EN-FR	63.0	82.5

Table 6: Model accuracy on the source and target language using original syntax. The source language scores for English vary between language pairs because of the difference in syntactic annotation and role subset used.

We also include the individual  $F_1$  scores for the top-10 most frequent labels for EN-CZ transfer with original syntax in table 7. The model provides meaningful predictions here, despite low overall accuracy.

Most of the labels<sup>2</sup> are self-explanatory: Patient (PAT), Actor (ACT), Time (TWHEN), Effect (EFF), Location (LOC), Manner (MANN), Addressee (ADDR), Extent (EXT). CPHR marks the

<sup>2</sup><http://ufal.mff.cuni.cz/~toman/pcedt/en/functors.html>

Label	Freq.	$F_1$	Re.	Pr.
PAT	14707	69.4	70.0	68.7
ACT	14303	81.1	81.7	80.4
TWHEN	3631	70.6	65.1	77.0
EFF	2601	45.4	67.2	34.3
LOC	1990	41.8	35.3	51.3
MANN	1208	54.0	63.8	46.9
ADDR	1045	30.2	34.4	26.8
CPHR	791	20.4	13.1	45.0
EXT	708	42.2	40.5	44.1
DIR3	695	20.1	17.3	23.9

Table 7: EN-CZ transfer (with original syntax),  $F_1$ , recall and precision for the top-10 most frequent roles.

nominal part of a complex predicate, as in “to have [a plan]<sub>CPHR</sub>”, and DIR3 indicates destination.

### 5.3 Additional Experiments

We now evaluate the contribution of different aspects of the feature representation to the performance of the model. Table 8 contains the results for English-French.

Features	Orig	Trans
POS	47.5	47.5
POS, Synt	53.0	53.1
POS, Cls	53.7	53.7
POS, Gloss	63.7	63.7
POS, Synt, Cls	55.9	56.4
POS, Synt, Gloss	65.2	66.3
POS, Cls, Gloss	61.5	61.5
POS, Synt, Cls, Gloss	63.0	65.1

Table 8: EN-FR model transfer accuracy with different feature subsets, using original and transferred syntactic information.

The fact that the model performs slightly better with transferred syntax may be explained by two factors. Firstly, as we already mentioned, the original syntactic annotation is also produced automatically. Secondly, in the model transfer setup it is more important how closely the syntactic-semantic interface on the target side resembles that on the source side than how well it matches the “true” structure of the target language, and in this respect a transferred dependency parser may have an advantage over one trained on target-language data.

The high impact of the Gloss features here

may be partly attributed to the fact that the mapping is derived from the same corpus as the evaluation data – Europarl (Koehn, 2005) – and partly by the similarity between English and French in terms of word order, usage of articles and prepositions. The moderate contribution of the cross-lingual cluster features are likely due to the insufficient granularity of the clustering for this task.

For more distant language pairs, the contributions of individual feature groups are less interpretable, so we only highlight a few observations. First of all, both EN-CZ and CZ-EN benefit noticeably from the use of the original syntactic annotation, including dependency relations, but not from the transferred syntax, most likely due to the low syntactic transfer performance. Both perform better when lexical information is available, although the improvement is not as significant as in the case of French – only up to 5%.

The situation with Chinese is somewhat complicated in that adding lexical information here fails to yield an improvement in terms of the metric considered. This is likely due to the fact that we consider only the core roles, which can usually be predicted with high accuracy based on syntactic information alone.

## 6 Related Work

Development of robust statistical models for core NLP tasks is a challenging problem, and adaptation of existing models to new languages presents a viable alternative to exhaustive annotation for each language. Although the models thus obtained are generally imperfect, they can be further refined for a particular language and domain using techniques such as active learning (Settles, 2010; Chen et al., 2011).

Cross-lingual annotation projection (Yarowsky et al., 2001) approaches have been applied extensively to a variety of tasks, including POS tagging (Xi and Hwa, 2005; Das and Petrov, 2011), morphology segmentation (Snyder and Barzilay, 2008), verb classification (Merlo et al., 2002), mention detection (Zitouni and Florian, 2008), LFG parsing (Wróblewska and Frank, 2009), information extraction (Kim et al., 2010), SRL (Padó and Lapata, 2009; van der Plas et al., 2011; Annesi and Basili, 2010; Tonelli and Pianta, 2008), dependency parsing (Naseem et al., 2012; Ganchev et al., 2009; Smith and Eisner, 2009; Hwa et al., 2005) or temporal relation pre-

diction (Spreyer and Frank, 2008). Interestingly, it has also been used to propagate morphosyntactic information between old and modern versions of the same language (Meyer, 2011).

Cross-lingual model transfer methods (McDonald et al., 2011; Zeman and Resnik, 2008; Durrett et al., 2012; Søgaard, 2011; Lopez et al., 2008) have also been receiving much attention recently. The basic idea behind model transfer is similar to that of cross-lingual annotation projection, as we can see from the way parallel data is used in, for example, McDonald et al. (2011).

A crucial component of direct transfer approaches is the unified feature representation. There are at least two such representations of lexical information (Klementiev et al., 2012; Täckström et al., 2012), but both work on word level. This makes it hard to account for phenomena that are expressed differently in the languages considered, for example the syntactic function of a certain word may be indicated by a preposition, inflection or word order, depending on the language. Accurate representation of such information would require an extra level of abstraction (Hajič, 2002).

A side-effect of using adaptation methods is that we are forced to use the same annotation scheme for the task in question (SRL, in our case), which in turn simplifies the development of cross-lingual tools for downstream tasks. Such representations are also likely to be useful in machine translation.

Unsupervised semantic role labeling methods (Lang and Lapata, 2010; Lang and Lapata, 2011; Titov and Klementiev, 2012a; Lorenzo and Cerisara, 2012) also constitute an alternative to cross-lingual model transfer.

For an overview of semi-supervised approaches we refer the reader to Titov and Klementiev (2012b).

## 7 Conclusion

We have considered the cross-lingual model transfer approach as applied to the task of semantic role labeling and observed that for closely related languages it performs comparably to annotation projection approaches. It allows one to quickly construct an SRL model for a new language without manual annotation or language-specific heuristics, provided an accurate model is available for one of the related languages along with a certain amount of parallel data for the two languages. While an-

notation projection approaches require sentence- and word-aligned parallel data and crucially depend on the accuracy of the syntactic parsing and SRL on the source side of the parallel corpus, cross-lingual model transfer can be performed using only a bilingual dictionary.

Unsupervised SRL approaches have their advantages, in particular when no annotated data is available for any of the related languages and there is a syntactic parser available for the target one, but the annotation they produce is not always sufficient. In applications such as Information Retrieval it is preferable to have precise labels, rather than just clusters of arguments, for example.

Also note that when applying cross-lingual model transfer in practice, one can improve upon the performance of the simplistic model we use for evaluation, for example by picking the features manually, taking into account the properties of the target language. Domain adaptation techniques can also be employed to adjust the model to the target language.

## Acknowledgments

The authors would like to thank Alexandre Klementiev and Ryan McDonald for useful suggestions and Täckström et al. (2012) for sharing the cross-lingual word representations. This research is supported by the MMCI Cluster of Excellence.

## References

- Omri Abend, Roi Reichart, and Ari Rappoport. 2009. Unsupervised argument identification for semantic role labeling. In *Proceedings of the Joint Conference of the 47<sup>th</sup> Annual Meeting of the ACL and the 4<sup>th</sup> International Joint Conference on Natural Language Processing of the AFNLP, ACL '09*, pages 28–36, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Paolo Annesi and Roberto Basili. 2010. Cross-lingual alignment of FrameNet annotations through hidden Markov models. In *Proceedings of the 11<sup>th</sup> international conference on Computational Linguistics and Intelligent Text Processing, CICLing'10*, pages 12–25, Berlin, Heidelberg. Springer-Verlag.
- Roberto Basili, Diego De Cao, Danilo Croce, Bonaventura Coppola, and Alessandro Moschitti. 2009. Cross-language frame semantics transfer in bilingual corpora. In Alexander F. Gelbukh, editor, *Proceedings of the 10<sup>th</sup> International Conference on Computational Linguistics and Intelligent Text Processing*, pages 332–345.
- Anders Björkelund, Love Hafdel, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 43–48, Boulder, Colorado, June. Association for Computational Linguistics.
- Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *ACL*.
- Chenhua Chen, Alexis Palmer, and Caroline Sporleder. 2011. Enhancing active learning for semantic role labeling via compressed dependency trees. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 183–191, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. *Proceedings of the Association for Computational Linguistics*.
- Greg Durrett, Adam Pauls, and Dan Klein. 2012. Syntactic transfer using a bilingual lexicon. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1–11, Jeju Island, Korea, July. Association for Computational Linguistics.
- Andreas Eisele and Yu Chen. 2010. MultiUN: A multilingual corpus from United Nation documents. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA).
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.
- Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proceedings of the 47<sup>th</sup> Annual Meeting of the ACL*, pages 369–377, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Qin Gao and Stephan Vogel. 2011. Corpus expansion for statistical machine translation with semantic role label substitution rules. In *Proceedings of the 49<sup>th</sup> Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 294–298, Portland, Oregon, USA.
- Trond Grenager and Christopher D. Manning. 2006. Unsupervised discovery of a statistical verb lexicon. In *Proceedings of EMNLP*.
- Jan Hajič. 2002. Tectogrammatical representation: Towards a minimal transfer in machine translation. In Robert Frank, editor, *Proceedings of the 6<sup>th</sup> International Workshop on Tree Adjoining Grammars*

- and Related Frameworks (TAG+6), pages 216–226, Venezia. Universita di Venezia.
- Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado.
- Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Ondřej Bojar, Silvie Cinková, Eva Fučková, Marie Mikulová, Petr Pajas, Jan Popelka, Jiří Semecký, Jana Šindlerová, Jan Štěpánek, Josef Toman, Zdeňka Urešová, and Zdeněk Žabokrtský. 2012. Announcing Prague Czech-English dependency treebank 2.0. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May. European Language Resources Association (ELRA).
- Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel text. *Natural Language Engineering*, 11(3):311–325.
- Richard Johansson and Pierre Nugues. 2008. Dependency-based semantic role labeling of PropBank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 69–78, Honolulu, Hawaii.
- Michael Kaiser and Bonnie Webber. 2007. Question answering based on semantic roles. In *ACL Workshop on Deep Linguistic Processing*.
- Seokhwan Kim, Minwoo Jeong, Jonghoon Lee, and Gary Geunbae Lee. 2010. A cross-lingual annotation projection approach for relation detection. In *Proceedings of the 23<sup>rd</sup> International Conference on Computational Linguistics, COLING '10*, pages 564–571, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Paul Kingsbury, Nianwen Xue, and Martha Palmer. 2004. Propbanking in parallel. In *Proceedings of the Workshop on the Amazing Utility of Parallel and Comparable Corpora, in conjunction with LREC'04*.
- Alexandre Klementiev, Ivan Titov, and Binod Bhatnagar. 2012. Inducing crosslingual distributed representations of words. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, Bombay, India.
- Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT.
- Joel Lang and Mirella Lapata. 2010. Unsupervised induction of semantic roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, California, June. Association for Computational Linguistics.
- Joel Lang and Mirella Lapata. 2011. Unsupervised semantic role induction via split-merge clustering. In *Proc. of Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of the 23<sup>rd</sup> International Conference on Computational Linguistics (Coling 2010)*, Beijing, China.
- Adam Lopez, Daniel Zeman, Michael Nossal, Philip Resnik, and Rebecca Hwa. 2008. Cross-language parser adaptation between related languages. In *IJCNLP-08 Workshop on NLP for Less Privileged Languages*, pages 35–42, Hyderabad, India, January.
- Alejandra Lorenzo and Christophe Cerisara. 2012. Unsupervised frame based semantic role induction: application to French and English. In *Proceedings of the ACL 2012 Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages*, pages 30–35, Jeju, Republic of Korea, July. Association for Computational Linguistics.
- Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pages 62–72, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Paola Merlo, Suzanne Stevenson, Vivian Tsang, and Gianluca Allaria. 2002. A multi-lingual paradigm for automatic verb classification. In *Proceedings of the 40<sup>th</sup> Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 207–214, Philadelphia, PA.
- Roland Meyer. 2011. New wine in old wineskins?—Tagging old Russian via annotation projection from modern translations. *Russian Linguistics*, 35(2):267(15).
- Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proceedings of the 50<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*, pages 629–637, Jeju Island, Korea, July. Association for Computational Linguistics.
- Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Comput. Linguist.*, 34(4):513–553, December.

- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1).
- Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36:307–340.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31:71–105.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of LREC*, May.
- Mark Sammons, Vinod Vydiswaran, Tim Vieira, Nikhil Johri, Ming wei Chang, Dan Goldwasser, Vivek Srikumar, Gourab Kundu, Yuancheng Tu, Kevin Small, Joshua Rule, Quang Do, and Dan Roth. 2009. Relation alignment for textual entailment recognition. In *Text Analysis Conference (TAC)*.
- Burr Settles. 2010. Active learning literature survey. *Computer Sciences Technical Report*, 1648.
- Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *EMNLP*.
- David A Smith and Jason Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 822–831. Association for Computational Linguistics.
- Benjamin Snyder and Regina Barzilay. 2008. Cross-lingual propagation for morphological analysis. In *Proceedings of the 23rd national conference on Artificial intelligence*.
- Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49<sup>th</sup> Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 2 of *HLT '11*, pages 682–686, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kathrin Spreyer and Anette Frank. 2008. Projection-based acquisition of a temporal labeller. *Proceedings of IJCNLP 2008*.
- Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 477–487, Montréal, Canada.
- Cynthia A. Thompson, Roger Levy, and Christopher D. Manning. 2003. A generative model for semantic role labeling. In *Proceedings of the 14<sup>th</sup> European Conference on Machine Learning*, ECML 2003, pages 397–408, Dubrovnik, Croatia.
- Ivan Titov and Alexandre Klementiev. 2012a. A Bayesian approach to unsupervised semantic role induction. In *Proc. of European Chapter of the Association for Computational Linguistics (EACL)*.
- Ivan Titov and Alexandre Klementiev. 2012b. Semi-supervised semantic role labeling: Approaching from an unsupervised perspective. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, Bombay, India, December.
- Sara Tonelli and Emanuele Pianta. 2008. Frame information transfer from English to Italian. In *Proceedings of LREC 2008*.
- Lonneke van der Plas, James Henderson, and Paola Merlo. 2009. Domain adaptation with artificial data for semantic parsing of speech. In *Proc. 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 125–128, Boulder, Colorado.
- Lonneke van der Plas, Paola Merlo, and James Henderson. 2011. Scaling up automatic cross-lingual semantic role annotation. In *Proceedings of the 49<sup>th</sup> Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, HLT '11, pages 299–304, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Alina Wróblewska and Anette Frank. 2009. Cross-lingual projection of LFG F-structures: Building an F-structure bank for Polish. In *Eighth International Workshop on Treebanks and Linguistic Theories*, page 209.
- Dekai Wu and Pascale Fung. 2009. Can semantic role labeling improve SMT? In *Proceedings of 13<sup>th</sup> Annual Conference of the European Association for Machine Translation (EAMT 2009)*, Barcelona.
- Chenhai Xi and Rebecca Hwa. 2005. A backoff model for bootstrapping resources for non-English languages. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 851–858, Stroudsburg, PA, USA.
- David Yarowsky, Grace Ngai, and Ricahrd Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of Human Language Technology Conference*.
- Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, pages 35–42, Hyderabad, India, January. Asian Federation of Natural Language Processing.
- Imed Zitouni and Radu Florian. 2008. Mention detection crossing the language barrier. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

# DERIVBASE: Inducing and Evaluating a Derivational Morphology Resource for German

Britta Zeller\*    Jan Šnajder†    Sebastian Padó\*

\*Heidelberg University, Institut für Computerlinguistik  
69120 Heidelberg, Germany

†University of Zagreb, Faculty of Electrical Engineering and Computing  
Unska 3, 10000 Zagreb, Croatia

{zeller, pado}@cl.uni-heidelberg.de    jan.snajder@fer.hr

## Abstract

Derivational models are still an under-researched area in computational morphology. Even for German, a rather resource-rich language, there is a lack of large-coverage derivational knowledge. This paper describes a rule-based framework for inducing *derivational families* (i.e., clusters of lemmas in derivational relationships) and its application to create a high-coverage German resource, DERIVBASE, mapping over 280k lemmas into more than 17k non-singleton clusters. We focus on the rule component and a qualitative and quantitative evaluation. Our approach achieves up to 93% precision and 71% recall. We attribute the high precision to the fact that our rules are based on information from grammar books.

## 1 Introduction

Morphological processing is generally recognized as an important step for many NLP tasks. Morphological analyzers such as lemmatizers and part of speech (POS) taggers are commonly the first NLP tools developed for any language (Koskenniemi, 1983; Brill, 1992). They are also applied in NLP applications where little other linguistic analysis is performed, such as linguistic annotation of corpora or terminology acquisition; see Daille et al. (2002) for an informative summary.

Most work on computational morphology has focused on inflectional morphology, that is, the handling of grammatically determined variation of form (Bickel and Nichols, 2001), which can be understood, overimplying somewhat, as a normalization step. Derivational morphology, which is concerned with the formation of new words from existing ones, has received less attention. Exam-

ples are nominalization (*to understand* → *the understanding*), verbalization (*the shelf* → *to shelve*), and adjectivization (*the size* → *sizable*). Part of the reason for the relative lack of attention lies in the morphological properties of English, such as the presence of many zero derivations (*the fish* → *to fish*), the dominance of suffixation, and the relative absence of stem changes in derivation. For these reasons, simple *stemming* algorithms (Porter, 1980) provide a cheap and accurate approximation to English derivation.

Two major NLP resources deal with derivation. WordNet lists so-called “morphosemantic” relations (Fellbaum et al., 2009) for English, and a number of proposals exist for extending WordNets in other languages with derivational relations (Bilgin et al., 2004; Pala and Hlaváčková, 2007). CatVar, the “Categorial Variation Database of English” (Habash and Dorr, 2003), is a lexicon aimed specifically at derivation. It groups English nouns, verbs, adjectives, and adverbs into derivational equivalence classes or *derivational families* such as

*ask<sub>V</sub> asker<sub>N</sub> asking<sub>N</sub> asking<sub>A</sub>*

Derivational families are commonly understood as groups of derivationally related lemmas (Daille et al., 2002; Milin et al., 2009). The lemmas in CatVar come from various open word classes, and multiple words may be listed for the same POS. The above family lists two nouns: an event noun (*asking*) and an agentive noun (*asker*). However, CatVar does not consider prefixation, which is why, e.g., the adjective *unasked* is missing.

CatVar has found application in different areas of English NLP. Examples are the acquisition of paraphrases that cut across POS lines, applied, for example, in textual entailment (Szpektor and Dagan, 2008; Berant et al., 2012). Then there is the induction and extension of semantic roles resources for predicates of various parts of speech (Meyers et al., 2004; Green et al., 2004). Finally, CatVar has

been used as a lexical resource to generate sentence intersections (Thadani and McKeown, 2011).

In this paper, we describe the project of obtaining derivational knowledge for German to enable similar applications. Even though there are two derivational resources for this language, IMSLEX (Fitschen, 2004) and CELEX (Baayen et al., 1996), both have shortcomings. The former does not appear to be publicly available, and the latter has a limited coverage (50k lemmas) and does not explicitly represent derivational relationships within families, which are necessary for fine-grained optimization of families. For this reason, we look into building a novel derivational resource for German. Unfortunately, the approach used to build CatVar cannot be adopted: it builds on a collection of high-quality lexical-semantic resources such as NOMLEX (Macleod et al., 1998), which are not available for German.

Instead, we employ a rule-based framework to define derivation rules that cover both suffixation and prefixation and describes stem changes. Following the work of Šnajder and Dalbelo Bašić (2010), we define the derivational processes using derivational rules and higher-order string transformation functions. The derivational rules induce a partition of the language’s lemmas into derivational families. Our method is applicable to many languages if the following are available: (1) a comprehensive set of lemmas (optionally including gender information); (2) knowledge about admissible derivational patterns, which can be gathered, for example, from linguistics textbooks.

The result is a freely available high-precision high-coverage resource for German derivational morphology that has a structure parallel to CatVar, but was obtained without using manually constructed lexical-semantic resources. We conduct a thorough evaluation of the induced derivational families both regarding precision and recall.

**Plan of the paper.** Section 2 discusses prior work. Section 3 defines our derivation model that is applied to German in Section 4. Sections 5 and 6 present our evaluation setup and results. Section 7 concludes the paper and outlines future work.

## 2 Related Work

Computational models of morphology have a long tradition. Koskenniemi (1983) was the first who analyzed and generated morphological phenomena computationally. His two-level theory has been

applied in finite state transducers (FST) for several languages (Karttunen and Beesley, 2005).

Many recent approaches automatically induce morphological information from corpora. They are either based solely on corpus statistics (Déjean, 1998), measure semantic similarity between input and output lemma (Schone and Jurafsky, 2000), or bootstrap derivation rules starting from seed examples (Piasecki et al., 2012). Hammarström and Borin (2011) give an extensive overview of state-of-the-art unsupervised learning of morphology. Unsupervised approaches operate at the level of word-forms and have complementary strengths and weaknesses to rule-based approaches. On the upside, they do not require linguistic knowledge; on the downside, they have a harder time distinguishing between derivation and inflection, which may result in lower precision, and are not guaranteed to yield analyses that correspond to linguistic intuition. An exception is the work by Gaussier (1999), who applies an unsupervised model to construct derivational families for French.

For German, several morphological tools exist. Morphix is a classification-based analyzer and generator of German words on the inflectional level (Finkler and Neumann, 1988). SMOR (Schmid et al., 2004) employs a finite-state transducer to analyze German words at the inflectional, derivational, and compositional level, and has been used in other morphological analyzers, e.g., Morphisto (Zielinski and Simon, 2008). The site *canoonet*<sup>1</sup> offers broad-coverage information about the German language including derivational word formation.

## 3 Framework

In this section, we describe our rule-based model of derivation, its operation to define derivational families, and the application of the model to German. We note that the model is purely surface-based, i.e., it does not model any semantic regularities beyond those implicit in string transformations. We begin by outlining the characteristics of German derivational morphology.

### 3.1 German Derivational Morphology

As German is a morphologically complex language, we analyzed its derivation processes before implementing our rule-based model. We relied on traditional grammar books and lexicons, e.g., Hoepfner (1980) and Augst (1975), in order to linguistically

<sup>1</sup><http://canoonet.net>

justify our assumptions as well as to achieve the best possible precision and coverage.

We concentrate on German derivational processes that involve nouns, verbs, and adjectives.<sup>2</sup> Nouns are simple to recognize due to capitalization: *stauen*<sub>V</sub> – *Stau*<sub>N</sub> (to jam – jam), *essen*<sub>V</sub> – *Essen*<sub>N</sub> (to eat – food). Verbs bear three typical suffixes (-en, -eln, -ern). An example of a derived verb is *fest*<sub>A</sub> – *festigen*<sub>V</sub> (tight – to tighten), where -ig is the derivational suffix. Adjectivization works similarly: *Tag*<sub>N</sub> – *täglich*<sub>A</sub> (day – daily).

This example shows that derivation can also involve stem changes in the form of umlaut (e.g., *a* → *ä*) and ablaut shift, e.g., *sieden*<sub>V</sub> – *Sud*<sub>N</sub> (to boil – infusion). Other frequent processes in German derivation are circumfixation (*Haft*<sub>N</sub> – *inhaftieren*<sub>V</sub> (arrest – to arrest)) and prefixation (*heben*<sub>V</sub> – *beheben*<sub>V</sub> (to raise – to remedy)). Prefixation often indicates a semantic shift, either in terms of the general meaning (as above) or in terms of the polarity (*klar*<sub>A</sub> – *unklar*<sub>A</sub> (clear – unclear)). Also note that affixes can be either Germanic, e.g., *ölen* – *Ölung* (to oil – oiling), or Latin/Greek, e.g., *generieren* – *Generator* (to generate – generator).

As this analysis shows, derivation in German involves transformation as well as affixation processes, which has to be taken into account when modeling a derivational resource.

### 3.2 A Rule-based Derivation Model

The purpose of a derivational model is to define a set of transformations that correspond to valid derivational word formation rules. Rule-based frameworks offer convenient representations for derivational morphology because they can take advantage of linguistic knowledge about derivation, have interpretable representations, and can be fine-tuned for high precision. The choice of the framework is in principle arbitrary, as long as it can conveniently express the derivational phenomena of a language. Typically used for this purpose are two-level formalism rules (Karttunen and Beesley, 1992) or XFST replace rules (Beesley and Karttunen, 2003).

In this paper, we adopt the modeling framework proposed by Šnajder and Dalbello Bašić (2010). The framework corresponds closely to simple, human-readable descriptions in traditional gram-

<sup>2</sup>We ignore adverb derivation; the German language distinguishes between adverbial adjectives and adverbs, the latter being a rather unproductive class and thus of no interest for derivation (Schiller et al., 1999).

mar books. The expressiveness of the formalism is equivalent to the replacement rules commonly used in finite state frameworks, thus the rules can be compiled into FSTs for efficient processing.

The framework makes a clear distinction between inflectional and derivational morphology and provides separate modeling components for these two; we only make use of the derivation modeling component. We use an implementation of the modeling framework in Haskell. For details, see the studies by Šnajder and Dalbello Bašić (2008) and Šnajder and Dalbello Bašić (2010).

The building blocks of the derivational component are *derivational rules (patterns)* and *transformation functions*. A derivational rule describes the derivation of a *derived* word from a *basis* word. A derivational rule *d* is defined as a triple:

$$d = (t, \mathcal{P}_1, \mathcal{P}_2) \quad (1)$$

where *t* is the transformation function that maps the word’s stem (or lemma) into the derived word’s stem (or lemma), while  $\mathcal{P}_1$  and  $\mathcal{P}_2$  are the sets of inflectional paradigms of the basis word and the derived word, respectively, which specify the morphological properties of the rule’s input and output. For German, our study assumes that inflectional paradigms are combinations of part-of-speech and gender information (for nouns).

A transformation function  $t : S \rightarrow \wp(S)$  maps strings to a set of strings, representing possible transformations. At the lowest level, *t* is defined in terms of atomic string replacement operations (replacement of prefixes, suffixes, and infixes). The framework then uses the notion of higher-order functions – functions that take other transformations as arguments and return new transformations as results – to succinctly define common derivational processes such as prefixation, suffixation, and stem change. More complex word-formation rules, such as those combining prefixation and suffixation, can be obtained straightforwardly by functional composition.

Table 1 summarizes the syntax we use for transformation functions and shows two example derivational rules. Rule 1 defines an English adjectivization rule. It uses the conditional *try* operator to apply to nouns with and without the -ion suffix (*action* – *active*, *instinct* – *instinctive*). Infix replacement is used to model stem alternation, as shown in rule 2 for German nominalization, e.g., *vermacht*<sub>A</sub> – *Vermächtnis*<sub>N</sub> (*bequeathed* – *bequest*).

Function	Description
$sfx(s)$	concatenate the suffix $s$
$dsfx(s)$	delete the suffix $s$
$aifx(s1, s2)$	alternate the infix $s1$ to $s2$
$try(t)$	perform transformation $t$ , if possible
$opt(t)$	optionally perform transformation $t$
$uml$	alternate infixes for an umlaut shift: $uml = aifx(\{(a, ä), (o, ö), (u, ü)\})$
Examples	
1 (EN)	$(sfx(ive) \circ try(dsfx(ion)), \mathcal{N}, \mathcal{A})$ “derive <i>-ive</i> adjectives from nouns potentially ending in <i>-ion</i> ”
2 (DE)	$(sfx(nis) \circ try(uml), \mathcal{A}, \mathcal{N})$ “derive <i>-nis</i> nouns from adjectives with optional umlaut creation”

Table 1: Transformation functions and exemplary derivational rules in the framework by Šnajder and Dalbelo Bašić (2010)

$\mathcal{N}$  and  $\mathcal{A}$  denote the paradigms for nouns (without gender restriction) and adjectives, respectively.

### 3.3 Induction of Derivational Families

Recall that our goal is to induce derivational families, that is, classes of derivationally related words. We define derivational families on the basis of derivational rules as follows.

Given a lemma-paradigm pair  $(l, p)$  as input, a single derivational rule  $d = (t, \mathcal{P}_1, \mathcal{P}_2)$  generates a set of possible derivations  $L_d(l, p) = \{(l_1, p_1), \dots, (l_n, p_n)\}$ , where  $p \in \mathcal{P}_1$  and  $p_i \in \mathcal{P}_2$  for all  $i$ . Given a set of derivational rules  $\mathcal{D}$ , we define a binary derivation relation  $\rightarrow_{\mathcal{D}}$  between two lemma-paradigm pairs that holds if the second pair can be derived from the first one as:

$$(l_1, p_1) \rightarrow_{\mathcal{D}} (l_2, p_2) \quad (2)$$

iff  $\exists d \in \mathcal{D}. (l_2, p_2) \in L_d(l_1, p_1)$

Let  $\mathcal{L}$  denote the set of lemma-paradigm pairs. The set of *derivational families defined by  $\mathcal{D}$  on  $\mathcal{L}$*  is given by the equivalence classes of the transitive, symmetric, and reflexive closure of  $\rightarrow_{\mathcal{D}}$  over  $\mathcal{L}$ .

Note that in addition to the quality of the rules, the properties of  $\mathcal{L}$  plays a central role in the quality of the induced families. High coverage of  $\mathcal{L}$  is important because the transitivity of  $\rightarrow_{\mathcal{D}}$  ranges only over lemmas in  $\mathcal{L}$ , so low coverage of  $\mathcal{L}$  may result in fragmented derivational families. However,  $\mathcal{L}$  should also not contain erroneous lemma-paradigm pairs. The reason is that the derivational rules only define *admissible* derivations, which need not be morphologically valid, and therefore routinely over-

generate;  $\mathcal{L}$  plays an important role in filtering out derivations that are not attested in the data.

## 4 Building the Resource

### 4.1 Derivational Rules

We implemented the derivational rules from Hoepfner (1980) for verbs, nouns, and adjectives, covering all processes described in Section 3.1 (zero derivation, prefixation, suffixation, circumfixation, and stem changes). We found many derivational patterns in German to be conceptually simple (e.g., verb-noun zero derivation) so that substantial coverage can already be achieved with very simple transformation functions. However, there are many more complex patterns (e.g., suffixation combined with optional stem changes) that in sum also affect a considerable number of lemmas, which required us to either implement low-coverage rules or generalize existing rules. In order to preserve precision as much as possible, we restricted rule application by using *try* instead of *opt*, and by using gender information from the noun paradigms (for example, some rules only apply to masculine nouns and produce female nouns). As a result, we end up with high-coverage rules, such as derivations of person-denoting nouns ( $Schule_N - Schüler_N$  (*school - pupil*)) as well as high-accuracy rules such as negation prefixes ( $Pol_N - Gegenpol_N$  (*pole - antipole*)).

Even though we did not focus on the explanatory relevance of rules, we found that the underlying modeling formalism, and the methodology used to develop the model, offer substantial linguistic plausibility in practice. We had to resort to heuristics mostly for words with derivational transformations that are motivated by Latin or Greek morphology and do not occur regularly in German, e.g.,  $selegieren_V - Selektion_N$  (*select - selection*).

In the initial development phase, we implemented 154 rules, which took about 22 person-hours. We then revised the rules with the aim of increasing both precision and recall. To this end, we constructed a development set comprised of a sample of 1,000 derivational families induced using our rules. On this set, we inspected the derivational families for false positives, identified the problematic rules, and identified unused and redundant rules. In order to identify the false negatives, we additionally sampled a list of 1,000 lemmas and used string distance measures (cf. Section 5.1) to retrieve the 10 most similar words for each lemma not

Process	N-N	N-A	N-V	A-A	A-V	V-V
Zero derivation	–	1	5	–	–	–
Prefixation	10	–	5	5	2	9
+ Stem change	–	–	3	–	1	–
Suffixation	15	35	20	1	14	–
+ Stem change	2	8	7	–	3	1
Circumfixation	–	–	1	–	–	–
+ Stem change	–	–	1	–	–	–
Stem change	–	–	7	–	–	2
<i>Total</i>	27	44	49	6	20	12

Table 2: Breakdown of derivation rules by category of the basis and the derived word

already covered by the derivational families. The refinement process took another 8 person-hours. It revealed three redundant rules and seven missing rules, leading us to a total of 158 rules.

Table 2 shows the distribution of rules with respect to the derivational processes they implement and the part of speech combinations for the basis and the derived words. All affixations occur both with and without stem changes, mostly umlaut shifts. Suffixation is by far the most frequently used derivation process, and noun-verb derivation is most diverse in terms of derivational processes.

We also estimated the reliability of derivational rules by analyzing the accuracy of each rule on the development set. We assigned each rule a confidence rating on a three-level scale: L3 – very reliable (high-accuracy rules), L2 – generally reliable, and L1 – less reliable (low-accuracy rules). We manually analyzed the correctness of rule applications for 100 derivational families of different size (counting 2 up to 114 lemmas), and assigned 55, 79, and 24 rules to L3, L2 and L1, respectively.

## 4.2 Data and Preprocessing

For an accurate application of nominal derivation rules, we need a lemma list with POS and gender information. We POS-tag and lemmatize SDEWAC, a large German-language web corpus from which boilerplate paragraphs, ungrammatical sentences, and duplicate pages were removed (Faaß et al., 2010). For POS tagging and lemmatization, we use TreeTagger (Schmid, 1994) and determine grammatical gender with the morphological layer of the MATE Tools (Bohnet, 2010). We treat proper nouns like common nouns.

We apply three language-specific filtering steps based on observations in Section 3.1. First, we discard non-capitalized nominal lemmas. Second, we deleted verbal lemmas not ending in verb suffixes.

Third, we removed frequently occurring erroneous comparative forms of adjectives (usually formed by adding *-er*, like *neuer / newer*) by checking for the presence of lemmas without *-er* (*neu / new*).

An additional complication in German concerns prefix verbs, because prefix is separated in tensed instances. For example, the 3rd person male singular of *aufhören* (*to stop*) is *er hört auf* (*he stops*). Since most prefixes double as prepositions, the correct lemmas can only be reconstructed by parsing. We parse the corpus using the MST parser (McDonald et al., 2006) and recover prefix verbs by searching for instances of the dependency relation labeled PTKVZ.

Since SDEWAC, as a web corpus, still contains errors, we only take into account lemmas that occur three times or more in the corpus. Considering the size of SDEWAC, we consider this as a conservative filtering step that preserves high recall and provides a comprehensive basis for evaluation. After preprocessing and filtering, we run the induction of the derivational families as explained in Section 3 to obtain the DERIVBASE resource.

## 4.3 Statistics on DERIVBASE

The preparation of the SDEWAC corpus as explained in Section 4.2 yields 280,336 lemmas, which we cover with our resource. We induced a total of 239,680 derivational families from this data, with 17,799 non-singletons and 221,881 singletons (most of them due to compound nouns). 11,039 of the families consist of two lemmas, while the biggest contains 116 lemmas (an overgenerated family). The biggest family with perfect precision (i.e., it contains only morphologically related lemmas) contains 40 lemmas, e.g., *halten<sub>V</sub>*, *erhalten<sub>V</sub>*, *Verhältnis<sub>N</sub>* (*to hold, to uphold, relation*), etc. For comparison, CatVar v2.1 contains only 82,676 lemmas in 13,368 non-singleton clusters and 38,604 singletons.

The following sample family has seven members across all three POSes and includes prefixation, suffixation, and infix umlaut shifts:

*taub<sub>A</sub>* (*numb<sub>A</sub>*), *Taubheit<sub>Nf</sub>* (*numbness<sub>N</sub>*), *betäuben<sub>V</sub>* (*to anesthetize<sub>V</sub>*), *Betäubung<sub>Nf</sub>* (*anesthesia<sub>N</sub>*), *betäubt<sub>A</sub>* (*anesthetized<sub>A</sub>*), *betäubend<sub>A</sub>* (*anesthetic<sub>A</sub>*), *Betäuben<sub>Nn</sub>* (*act of anesthetizing<sub>N</sub>*)

## 5 Evaluation

### 5.1 Baselines

We use two baselines against which we compare the induced derivational families: (1) clusters obtained with the German version of Porter’s stemmer (Porter, 1980)<sup>3</sup> and (2) clusters obtained using string distance-based clustering. We have considered a number of string distance measures and tested them on the development set (cf. Section 4.1). The measure proposed by Majumder et al. (2007) turned out to be the most effective in capturing suffixal variation. For words  $X$  and  $Y$ , it is defined as

$$D_4(X, Y) = \frac{n - m + 1}{n + 1} \sum_{i=m}^n \frac{1}{2^{i-m}} \quad (3)$$

where  $m$  is the position of left-most character mismatch, and  $n + 1$  is the length of the longer of the two strings. To capture prefixal variation and stem changes, we use the  $n$ -gram based measure proposed by Adamson and Boreham (1974):

$$Dice_n(X, Y) = 1 - \frac{2c}{x + y} \quad (4)$$

where  $x$  and  $y$  are the total number of distinct  $n$ -grams in  $X$  and  $Y$ , respectively, and  $c$  is the number of distinct  $n$ -grams shared by both words. In our experiments, the best performance was achieved with  $n = 3$ .

We used hierarchical agglomerative clustering with average linkage. To reduce the computational complexity, we performed a preclustering step by recursively partitioning the set of lemmas sharing the same prefix into partitions of manageable size (1000 lemmas). Initially, we set the number of clusters to be roughly equal to the number of induced derivational families. For the final evaluation, we optimized the number of clusters based on  $F_1$  score on calibration and validation sets (cf. Section 5.3).

### 5.2 Evaluation Methodology

The induction of derivational families could be evaluated globally as a clustering problem. Unfortunately, cluster evaluation is a non-trivial task for which there is no consensus on the best approach (Amigó et al., 2009). We decided to perform our evaluation at the level of pairs: we manually judge for a set of pairs whether they are derivationally related or not.

<sup>3</sup><http://snowball.tartarus.org>

We obtain the gold standard for this evaluation by sampling lemmas from the lemma list. With random sampling, the evaluation would be unrealistic because a vast majority of pairs would be derivationally unrelated and count as true negatives in our analysis. Moreover, in order to reliably estimate the overall precision of the obtained derivational families, we need to evaluate on pairs sampled from these families. On the other hand, in order to assess recall, we need to sample from pairs that are not included in our derivational families.

To obtain reliable estimates of both precision and recall, we decided to draw two different samples: (1) a sample of lemma pairs sampled from the induced derivational families, on which we estimate precision (*P-sample*) and (2) a sample of lemma pairs sampled from the set of possibly derivationally related lemma pairs, on which we estimate recall (*R-sample*). In both cases, pairs  $(l_1, l_2)$  are sampled in two steps: first a lemma  $l_1$  is drawn from a non-singleton family, then the second lemma  $l_2$  is drawn from the derivational family of  $l_1$  (*P-sample*) or the set of lemmas possibly related to  $l_1$  (*R-sample*). The set of possibly related lemmas is a union of the derivational family of  $l_1$ , the clusters of  $l_1$  obtained with the baseline methods, and  $k$  lemmas most similar to  $l_1$  according to the two string distance measures. We use  $k = 7$  in our experiments. This is based on preliminary experiments on the development set (cf. Section 4.1), which showed that  $k = 7$  retrieves about 92% of the related lemmas retrieved for  $k = 20$  with a much smaller number of true negatives. Thus, the evaluation on the *R-sample* might overestimate the recall, but only slightly so, while the *P-sample* yields a reliable estimate of precision by reducing the number of true negatives in the sample.

Both samples contain 2400 lemma pairs each. Lemmas included in the development set (Section 4.1) were excluded from sampling.

### 5.3 Gold Standard Annotation

Two German native speakers annotated the pairs from the *P-sample* and *R-samples*. We defined five categories into which all lemma pairs are classified as shown in Table 3. We count *R* and *M* as positives and *N*, *C*, *L* as negatives (cf. Section 3).<sup>4</sup> Note that this binary distinction would be sufficient to compute recall and precision. However, the more

<sup>4</sup>Ambiguous lemmas are categorized as positive (*R* or *M*) if there is a matching sense.

Label	Description	Example
<b>R</b>	$l_1$ and $l_2$ are morphologically and semantically related	<i>kratzig<sub>A</sub> – verkratz<sub>A</sub></i> ( <i>scratchy – scuffed</i> )
<b>M</b>	$l_1$ and $l_2$ are morphologically but not semantically related	<i>bomben<sub>V</sub> – bombig<sub>A</sub></i> ( <i>to bomb – smashing</i> )
<b>N</b>	no morphological relation	<i>belebt<sub>A</sub> – loben<sub>V</sub></i> ( <i>lively – to praise</i> )
<b>C</b>	no derivational relation, but the pair is compositionally related	<i>Filmende<sub>N</sub> – filmen<sub>V</sub></i> ( <i>end of film – to film</i> )
<b>L</b>	not a valid lemma (mis-lemmatization, wrong gender, foreign words)	<i>Haufe<sub>N</sub> – Häufung<sub>N</sub></i> ( <i>N/A – accumulation</i> )

Table 3: Categories for lemma pair classification

	Agreement	Cohen’s $\kappa$
R-sample	0.85	0.79
P-sample	0.86	0.70

Table 4: Inter-annotator agreement on validation sample

fine-grained five-class annotation scheme provides a more detailed picture. The separation between R and M gives a deeper insight into the semantics of the derivational families. Distinguishing between C and N, in turn, allows us to identify the pairs that are derivationally unrelated, but compositionally related, e.g., *Ehemann<sub>N</sub> – Ehefrau<sub>N</sub>* (*husband – wife*).

We first carried out a calibration phase in which the annotators double-annotated 200 pairs from each of the two samples and refined the annotation guidelines. In a subsequent validation phase, we computed inter-annotator agreements on the annotations of another 200 pairs each from the P- and the R-samples. Table 4 shows the proportion of identical annotations by both annotators as well as Cohen’s  $\kappa$  score (Cohen, 1968). We achieve substantial agreement for  $\kappa$  (Carletta, 1996). On the P-sample,  $\kappa$  is a little lower because the distribution of the categories is skewed towards R, which makes an agreement by chance more probable.

In our opinion, the IAA results were sufficiently high to switch to single annotation for the production phase. Here, each annotator annotated another 1000 pairs from the P-sample and R-sample so that the final test set consists of 2000 pairs from each sample. The P-sample contains 1663 positive (R+M) and 337 negative (N+C+L) pairs, respectively, the R-sample contains 575 positive and 1425 negative pairs. As expected, there are more positive

Method	Precision		Recall	
	P-sample	R-sample	P-sample	R-sample
DERIVBASE (initial)	0.83		0.58	
DERIVBASE-L123	0.83		<b>0.71</b>	
DERIVBASE-L23	0.88		0.61	
DERIVBASE-L3	<b>0.93</b>		0.35	
		R-sample		
Stemming	0.66		0.07	
String distance $D_4$	0.36		0.20	
String distance $Dice_3$	0.23		0.23	

Table 5: Precision and recall on test samples

pairs in the P-sample and more negative pairs in the R-sample.

## 6 Results

### 6.1 Quantitative Evaluation

Table 5 presents the overall results. We evaluate four variants of the induced derivational families: those obtained before rule refinement (DERIVBASE initial), and three variants after rule refinement: using all rules (DERIVBASE-L123), excluding the least reliable rules (DERIVBASE-L23), and using only highly reliable rules (DERIVBASE-L3).

We measure the precision of our method on the P-sample and recall on the R-sample. For the baselines, precision was also computed on the R-sample (computing it on P-sample, which is obtained from the induced derivational families, would severely underestimate the number of false positives). We omit the  $F_1$  score because its use for precision and recall estimates from different samples is unclear.

DERIVBASE reaches 83% precision when using all rules and 93% precision when using only highly reliable rules. DERIVBASE-L123 achieves the highest recall, outperforming other methods and variants by a large margin. Refinement of the initial model has produced a significant improvement in recall without losses in precision. The baselines perform worse than our method: the stemmer we use is rather conservative, which fragments the families and leads to a very low recall. The string distance-based approaches achieve more balanced precision and recall scores. Note that for these methods, precision and recall can be traded off against each other by varying the number of clusters; we chose the number of clusters by optimizing the  $F_1$  score on the calibration and validation sets.

All subsequent analyses refer to DERIVBASE-

Coverage	Accuracy		
	High	Low	Total
High	18	–	18
Low	53	21	74
<i>Total</i>	71	21	92

Table 6: Proportions of accuracy and coverage for direct derivations (measured on P-sample)

	P	R	P	R
N-N	0.78	0.68	N-A	<b>0.89</b> <b>0.83</b>
A-A	0.87	0.70	N-V	0.79 0.68
V-V	0.55	0.24	A-V	0.88 0.73

Table 7: Precision and recall across different part of speech (first POS: basis; second POS: derived word)

L123, which is the model with the highest recall. If optimal precision is required, DERIVBASE-L3 should however be preferred.

**Analysis by frequency.** We cross-classified our rules according to high/low accuracy and high/low coverage based on the pairs in the P-sample. We only considered directly derivationally related ( $\rightarrow_D$ ) pairs and defined “high accuracy” and “high coverage” as all rules above the 25th percentile in terms of accuracy and coverage, respectively. The results are shown in Table 6: all high-coverage rules are also highly accurate. Most rules are accurate but infrequent. Only 21 rules have a low accuracy, but all of them apply infrequently.

**Analysis by parts of speech.** Table 7 shows precision and recall values for different part of speech combinations for the basis and derived words. High precision and recall are achieved for N-A derivations. The recall is lowest for V-V derivations, suggesting that the derivational phenomena for this POS combination are not yet covered satisfactorily.

## 6.2 Error analysis

Table 8 shows the frequencies of true positives and false positives on the P-sample and false negatives on the R-sample for each annotated category. True negatives are not reported, since their analysis gives no deeper insight.

**True positives.** In our analysis we treated both R and M pairs as related, but it is interesting to see how many of the true positives are in fact semantically unrelated. Out of 1,663 pairs, 90% are semantically as well as morphologically related (R), e.g.,

Label	TPs	FPs	FNs
	P-sample	P-sample	R-sample
R	1,492	–	107
M	171	–	60
N	–	216	–
C	–	7	–
L	–	114	–
<i>Total</i>	1,663	337	167

Table 8: Predictions over annotated categories

*alkoholisieren<sub>V</sub> – antialkoholisch<sub>A</sub> (to alcoholize – nonalcoholic), Beschuldigung<sub>N</sub> – unschuldig<sub>A</sub> (accusation – innocent).* Most R pairs result from high-accuracy rules, i.e., zero derivation, negation prefixation and simple suffixation. The remaining 10% are only morphologically related (M), e.g., *beschwingt<sub>A</sub> – schwingen<sub>V</sub> (cheerful – to swing), Stolzieren<sub>N</sub> – stolz<sub>A</sub> (strut – proud).* In both pairs, the two lemmas share a common semantic concept – i.e., being in motion or being proud – but nowadays meanings have grown apart from each other. Among the M true positives, we observe prefixation derivations in 66% of the cases, often involving prefixation at both lemmas, e.g., *Erdenkliche<sub>N</sub> – bedenklich<sub>A</sub> (imaginable – questionable).*

**False positives.** We observe many errors in pairs involving short lemmas, e.g., *Gen<sub>N</sub> – genieren<sub>V</sub> (gene – to be embarrassed)*, where orthographic context is insufficient to reject the derivation. About 64% of the 337 incorrect pairs are of class N (unrelated lemmas). For example, the rule for deriving nouns denoting a male person incorrectly links *Morse<sub>N</sub> – Mörser<sub>N</sub> (Morse – mortar)*. Transitively applied rules often produce incorrect pairs; e.g., *Speiche<sub>N</sub> – speicherbar<sub>A</sub> (spoke – storable)* results from the rule chain *Speiche<sub>N</sub> → Speicher<sub>N</sub> → speichern<sub>V</sub> → speicherbar<sub>A</sub> (spoke → storage → to store → storable)*. Chains that involve ablaut shifts (cf. Section 3.1) can lead to surprising results, e.g., *Erringung<sub>N</sub> – rangiert<sub>A</sub> (achievement – shunted)*. Meanwhile, some pairs judged as unrelated by the annotators might conceivably be weakly related, such as *schlürfen<sub>V</sub> and schlurfen<sub>V</sub> (to sip – to shuffle)*, both of which refer to specific long drawn out sounds. About 20% out of these unrelated lemma pairs is due to derivations between proper nouns (PNs) and common nouns. This happens especially for short PNs (cf. the above example of *Morse*). However, since PNs also participate in valid derivations (e.g., *Chaplin – chaplinesque*),

one could investigate their impact on derivations rather than omitting them.

Errors of the category L – 34% of the false positives – are caused during preprocessing by the lemmatizer. They cannot be blamed on our derivational model, but of course form part of the output.

**False negatives.** Errors of this type are due to missing derivation rules, erroneous rules that leave some lemmas undiscovered, or the absence of lemmas in the corpus required for transitive closure. About 64% of the 167 missed pairs are of category R. About half of these pairs result from a lack of prefixation rules – mainly affecting verbs – with a wide variety of prefixes (*zu-*, *um-*, etc.), including prepositional prefixes like *herum-* (*around*) or *über-* (*over*). We intentionally ignored these derivations, since they frequently lead to semantically unrelated pairs. In fact, merely five of the remaining 36% false negative pairs (M) do not involve prefixation. However, this analysis as well as the rather low coverage for verb-involved rules (cf. Table 7) shows that DERIVBASE might benefit from more prefix rules. Apart from the lack of prefixation coverage and a few other, rather infrequent rules, we did not find any substantial deficits. Most of the remaining errors are due to German idiosyncrasies and exceptional derivations, e.g., *fahren<sub>V</sub>* – *Fahrt<sub>N</sub>* (*drive* – *trip*), where the regular zero derivation would result in *Fahr*.

## 7 Conclusion and Future Work

In this paper, we present DERIVBASE, a derivational resource for German based on a rule-based framework. A few work days were enough to build the underlying rules with the aid of grammar textbooks. We collected derivational families for over 280,000 lemmas with high accuracy as well as solid coverage. The resource is freely available.<sup>5</sup>

Our approach for compiling a derivational resource is not restricted to German. In addition to the typologically most similar Germanic and Romance languages, it is also applicable to agglutinative languages like Finnish, or other fusional languages like Russian. Its main requirements are a large list of lemmas for the language (optionally with further morphological features) and linguistic literature on morphological patterns.

We have employed an evaluation method that uses two separate samples to assess precision and

recall to deal with the high number of false negatives. Our analyses indicate two interesting directions for future work: (a) specific handling of proper nouns, which partake in specific derivations; and (b) the use of graph clustering instead of the transitive closure to avoid errors resulting from long transitive chains.

Finally, we plan to employ distributional semantics methods (Turney and Pantel, 2010) to help remove semantically unrelated pairs as well as distinguish automatically between only morphologically (M) or both morphologically and semantically (R) related pairs. Last, but not least, this allows us to group derivation rules according to their semantic properties. For example, nouns with *-er* suffixes often denote persons and are agentivizations of a basis word (Bilgin et al., 2004).

## Acknowledgments

The first and third authors were supported by the EC project EXCITEMENT (FP7 ICT-287923). The second author was supported by the Croatian Science Foundation (project 02.03/162: “Derivational Semantic Models for Information Retrieval”). We thank the reviewers for their constructive comments.

## References

- George W. Adamson and Jillian Boreham. 1974. The use of an association measure based on character structure to identify semantically related pairs of words and document titles. *Information Processing and Management*, 10(7/8):253–260.
- Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486.
- Gerhard Augst. 1975. *Lexikon zur Wortbildung*. Forschungsberichte des Instituts für Deutsche Sprache. Narr, Tübingen.
- Harald R. Baayen, Richard Piepenbrock, and Leon Gulikers. 1996. *The CELEX Lexical Database. Release 2. LDC96L14*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA.
- Kenneth R Beesley and Lauri Karttunen. 2003. *Finite state morphology*, volume 18. CSLI publications Stanford.
- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2012. Learning entailment relations by global graph structure optimization. *Computational Linguistics*, 38(1):73–111.

<sup>5</sup><http://goo.gl/7KG2U>; license cc-by-sa 3.0

- Balthazar Bickel and Johanna Nichols. 2001. Inflectional morphology. In Timothy Shopen, editor, *Language Typology and Syntactic Description, Volume III: Grammatical categories and the lexicon*, pages 169–240. CUP, Cambridge.
- Orhan Bilgin, Özlem Çetinoğlu, and Kemal Oflazer. 2004. Morphosemantic relations in and across Wordnets. In *Proceedings of the Global WordNet Conference*, pages 60–66, Brno, Czech Republic.
- Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 89–97, Beijing, China.
- Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Workshop on Speech and Natural Language*, pages 112–116, Harriman, New York.
- Jean C. Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254.
- Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70:213–220.
- Béatrice Daille, Cécile Fabre, and Pascale Sébillot. 2002. Applications of computational morphology. In Paul Boucher, editor, *Many Morphologies*, pages 210–234. Cascadilla Press.
- Hervé Déjean. 1998. Morphemes as necessary concept for structures discovery from untagged corpora. In *Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning*, pages 295–298, Sydney, Australia.
- Gertrud Faaß, Ulrich Heid, and Helmut Schmid. 2010. Design and application of a gold standard for morphological analysis: SMOR in validation. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 803–810.
- Christiane Fellbaum, Anne Osherson, and Peter Clark. 2009. Putting semantics into WordNet's "morphosemantic" links. In *Proceedings of the Third Language and Technology Conference*, pages 350–358, Poznań, Poland.
- Wolfgang Finkler and Günter Neumann. 1988. Morphix - a fast realization of a classification-based approach to morphology. In *Proceedings of 4th Austrian Conference of Artificial Intelligence*, pages 11–19, Vienna, Austria.
- Arne Fitschen. 2004. *Ein computerlinguistisches Lexikon als komplexes System*. Ph.D. thesis, IMS, Universität Stuttgart.
- Éric Gaussier. 1999. Unsupervised learning of derivational morphology from inflectional lexicons. In *ACL'99 Workshop Proceedings on Unsupervised Learning in Natural Language Processing*, pages 24–30, College Park, Maryland, USA.
- Rebecca Green, Bonnie J. Dorr, and Philip Resnik. 2004. Inducing frame semantic verb classes from wordnet and Idoce. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 375–382, Barcelona, Spain.
- Nizar Habash and Bonnie Dorr. 2003. A categorial variation database for English. In *Proceedings of the Annual Meeting of the North American Association for Computational Linguistics*, pages 96–102, Edmonton, Canada.
- Harald Hammarström and Lars Borin. 2011. Unsupervised learning of morphology. *Computational Linguistics*, 37(2):309–350.
- Wolfgang Hoepfner. 1980. *Derivative Wortbildung der deutschen Gegenwartssprache und ihre algorithmische Analyse*. Narr, Tübingen.
- Lauri Karttunen and Kenneth R. Beesley. 1992. *Two-level rule compiler*. Xerox Corporation, Palo Alto Research Center.
- Lauri Karttunen and Kenneth R. Beesley. 2005. Twenty-five years of finite-state morphology. In Antti Arppe, Lauri Carlson, Krister Lindén, Jussi Pitulainen, Mickael Suominen, Martti Vainio, Hanna Westerlund, and Anssi Yli-Jyr, editors, *Inquiries into Words, Constraints and Contexts. Festschrift for Kimmo Koskenniemi on his 60th Birthday*, pages 71–83. CSLI Publications, Stanford, California.
- Kimmo Koskenniemi. 1983. *Two-level Morphology: A General Computational Model for Word-Form Recognition and Production*. Ph.D. thesis, University of Helsinki.
- Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. 1998. NOMLEX: A lexicon of nominalizations. In *In Proceedings of Euralex98*, pages 187–193.
- Prasenjit Majumder, Mandar Mitra, Swapan K. Parui, Gobinda Kole, Pabitra Mitra, and Kalyankumar Datta. 2007. YASS: Yet another suffix stripper. *ACM Transactions on Information Systems*, 25(4):18:1–18:20.
- Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *In Proceedings of the Conference on Computational Natural Language Learning*, pages 216–220, New York, NY.
- Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. Annotating noun argument structure for NomBank. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

- Petar Milin, Victor Kuperman, Aleksandar Kostic, and R Harald Baayen. 2009. Paradigms bit by bit: An information theoretic approach to the processing of paradigmatic structure in inflection and derivation. *Analogy in grammar: Form and acquisition*, pages 214–252.
- Karel Pala and Dana Hlaváčková. 2007. Derivational relations in Czech WordNet. In *Proceedings of the ACL Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies*, pages 75–81.
- Maciej Piasecki, Radoslaw Ramocki, and Marek Maziarz. 2012. Recognition of Polish derivational relations based on supervised learning scheme. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 916–922, Istanbul, Turkey.
- Martin Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.
- Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. 1999. Guidelines für das Tagging deutscher Textcorpora mit STTS. Technical report, Institut für maschinelle Sprachverarbeitung, Stuttgart.
- Helmut Schmid, Arne Fitschen, and Ulrich Heid. 2004. Smor: A German computational morphology covering derivation, composition and inflection. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, Lisbon, Portugal.
- Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.
- Patrick Schone and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of the Conference on Natural Language Learning*, pages 67–72, Lisbon, Portugal.
- Jan Šnajder and Bojana Dalbelo Bašić. 2008. Higher-order functional representation of Croatian inflectional morphology. In *Proceedings of the 6th International Conference on Formal Approaches to South Slavic and Balkan Languages*, pages 121–130, Dubrovnik, Croatia.
- Jan Šnajder and Bojana Dalbelo Bašić. 2010. A computational model of Croatian derivational morphology. In *Proceedings of the 7th International Conference on Formal Approaches to South Slavic and Balkan Languages*, pages 109–118, Dubrovnik, Croatia.
- Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 849–856, Manchester, UK.
- Kapil Thadani and Kathleen McKeown. 2011. Towards strict sentence intersection: Decoding and evaluation strategies. In *Proceedings of the ACL Workshop on Monolingual Text-To-Text Generation*, pages 43–53, Portland, Oregon.
- Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.
- Andrea Zielinski and Christian Simon. 2008. Morphisto - an open source morphological analyzer for German. In *Proceedings of the 7th International Workshop on Finite-State Methods and Natural Language Processing*, pages 224–231, Ispra, Italy.

# Crowdsourcing Interaction Logs to Understand Text Reuse from the Web

Martin Potthast   Matthias Hagen   Michael Völske   Benno Stein

Bauhaus-Universität Weimar  
99421 Weimar, Germany

<first name>.<last name>@uni-weimar.de

## Abstract

We report on the construction of the Webis text reuse corpus 2012 for advanced research on text reuse. The corpus compiles manually written documents obtained from a completely controlled, yet representative environment that emulates the web. Each of the 297 documents in the corpus is about one of the 150 topics used at the TREC Web Tracks 2009–2011, thus forming a strong connection with existing evaluation efforts. Writers, hired at the crowdsourcing platform oDesk, had to retrieve sources for a given topic and to reuse text from what they found. Part of the corpus are detailed interaction logs that consistently cover the search for sources as well as the creation of documents. This will allow for in-depth analyses of *how text is composed* if a writer is at liberty to reuse texts from a third party—a setting which has not been studied so far. In addition, the corpus provides an original resource for the evaluation of text reuse and plagiarism detectors, where currently only less realistic resources are employed.

## 1 Introduction

The web has become one of the most common sources for text reuse. When reusing text from the web, humans may follow a three step approach shown in Figure 1: searching for appropriate sources on a given topic, copying of text from selected sources, modification and paraphrasing of the copied text. A considerable body of research deals with the detection of text reuse, and, in particular, with the detection of cases of plagiarism (i.e., the reuse of text with the intent of disguising the fact that text has been reused). Similarly, a large number of commercial software systems is

being developed whose purpose is the detection of plagiarism. Both the developers of these systems as well as researchers working on the subject matter frequently claim their approaches to be searching the entire web or, at least, to be scalable to web size. However, there is hardly any evidence to substantiate this claim—rather the opposite can be observed: commercial plagiarism detectors have not been found to reliably identify plagiarism from the web (Köhler and Weber-Wulff, 2010), and the evaluation of research prototypes even under laboratory conditions shows that there is still a long way to go (Potthast et al., 2010b). We explain the disappointing state of the art by the lack of realistic, large-scale evaluation resources.

With our work, we want to contribute to closing the gap. In this regard the paper in hand introduces the Webis text reuse corpus 2012 (Webis-TRC-12), which, for the first time, emulates the *entire process* of reusing text from the web, both at scale and in a controlled environment. The corpus comprises a number of features that set it apart from previous ones: (1) the topic of each document in the corpus is derived from a topic of the TREC Web Track, and the sources to copy from have been retrieved manually from the ClueWeb corpus. (2) The search for sources is logged, including click-through and browsing data. (3) A fine-grained edit history has been recorded for each document. (4) A total of 297 documents were written with an average length of about 5700 words, whereas diversity is ensured via crowdsourcing. Altogether, this corpus forms the current most realistic sample of writers reusing text. The corpus is publicly available.<sup>1</sup>

### 1.1 Related Work

As organizers of the annual PAN plagiarism detection competitions,<sup>2</sup> we have introduced the first standardized evaluation framework for that pur-

<sup>1</sup><http://www.webis.de/research/corpora>

<sup>2</sup><http://pan.webis.de>

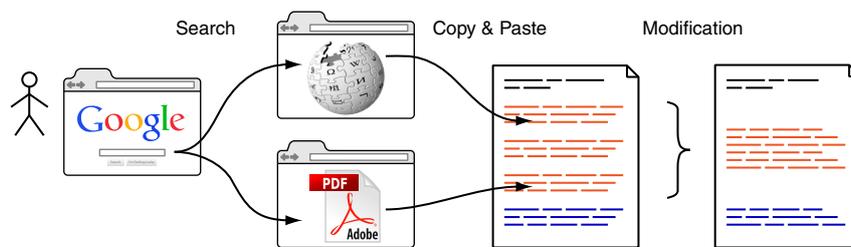


Figure 1: The basic steps of reusing text from the web (Potthast, 2011).

pose (Potthast et al., 2010b). Among others, it comprises a series of corpora that consist of automatically generated cases of plagiarism, provided in the form of the PAN plagiarism corpora 2009-2011. The corpora have been used to evaluate dozens of plagiarism detection approaches within the respective competitions in these years;<sup>3</sup> but even though they have been adopted by the community, a number of shortcomings render them less realistic:

1. All plagiarism cases were generated by randomly selecting text passages from documents and inserting them at random positions in a host document. This way, the reused passages do not match the topic of the host document.
2. The majority of the reused passages were modified in order to obfuscate the reuse. However, the applied modification strategies, again, are basically random: shuffling, replacing, inserting, or deleting words randomly. An effort was made to avoid non-readable text, yet none of it bears any semantics.
3. The corpus documents are parts of books from the Project Gutenberg. Many of these books are pretty old, whereas today the web is the predominant source for text reuse.

To overcome the second issue, about 4 000 passages were rewritten manually via crowdsourcing on Amazon’s Mechanical Turk for the 2011 corpus. But, because of the first issue (random passage insertion), a topic drift analysis can spot a reused passage more easily than a search within the document set containing the original source (Potthast et al., 2011). From these observations it becomes clear that there are limits for the automatic construction of such kinds of corpora. The Webis text reuse corpus 2012 addresses all of the mentioned issues since it has been constructed manually.

<sup>3</sup>See (Potthast et al., 2009; Potthast et al., 2010a; Potthast et al., 2011) for overviews of approaches and evaluation results of each competition.

Besides the PAN corpora, there are two other corpora that comprise “genuinely reused” text: the Clough09 corpus, and the Meter corpus. The former corpus consists of 57 answers to one of five computer science questions that were reused from a respective Wikipedia article (Clough and Stevenson, 2011). While the text was genuinely written by a number of volunteer students, the choice of topics is narrow, and text lengths range from 200 to 300 words, which is hardly more than 2-3 paragraphs. Also, the sources from which text was reused were given up front, so that there is no data about their retrieval. The Meter corpus annotates 445 cases of text reuse among 1 716 news articles (Clough et al., 2002). The cases of text reuse in this corpus are realistic for the news domain; however, they have not been created by the reuse process outlined in Figure 1. Note that in the news domain, text is often reused directly from a news wire without the need for retrieval. Our new corpus complements these two resources.

## 2 Corpus Construction

Two data sets form the basis for constructing our corpus, namely (1) a set of topics to write about and (2) a set of web pages to research about a given topic. With regard to the former, we resort to topics used at TREC, specifically to those used at the Web Tracks 2009–2011. With regard to the latter, we employ the ClueWeb corpus from 2009<sup>4</sup> (and not the “web in the wild”). The ClueWeb comprises more than one billion documents from ten languages and can be considered as a representative cross-section of the real web. It is a widely accepted resource among researchers and became one of the primary resources to evaluate the retrieval performance of search engines within several TREC tracks. Our corpus’s strong connection to TREC will allow for unforeseen synergies. Based on these decisions our

<sup>4</sup><http://lemurproject.org/clueweb09>

corpus construction steps can be summarized as follows:

1. Rephrasing of the 150 topics used at the TREC Web Tracks 2009–2011 so that they explicitly invite people to write an essay.
2. Indexing of the ClueWeb corpus category A (the entire English portion with about 0.5 billion documents) using the BM25F retrieval model plus additional features.
3. Development of a search interface that allows for answering queries within milliseconds and that is designed along the lines of commercial search interfaces.
4. Development of a browsing API for the ClueWeb, which serves ClueWeb pages on demand and which rewrites links of delivered pages, now pointing to their corresponding ClueWeb pages on our servers (instead of to the originally crawled URL).
5. Recruiting 27 writers, 17 of whom with a professional writing background, hired at the crowdsourcing platform oDesk from a wide range of hourly rates for diversity.
6. Instructing the writers to write one essay at a time of at least 5000 words length (corresponding to an average student’s homework assignment) about an open topic of their choice, using our search engine—hence browsing only ClueWeb pages.
7. Logging all writers’ interactions with the search engine and the ClueWeb on a per-essay basis at our site.
8. Logging all writers’ edits to their essays in a fine-grained edit log: a snapshot was taken whenever a writer stopped writing for more than 300ms.
9. Double-checking all of the essays for quality.

After having deployed the search engine and completed various usability tests, the actual corpus construction took nine months, from April 2012 through December 2012.

Obviously, the outlined experimental setup can serve different lines of research and is publicly available as well. The remainder of the section presents elements of our setup in greater detail.

## 2.1 Topic Preparation

Since the topics used at the TREC Web Tracks were not amenable for our purpose as is, we rephrased them so that they ask for writing an essay instead of searching for facts. Consider for example topic 001 of the TREC Web Track 2009:

*Query.* obama family tree

*Description.* Find information on President Barack Obama’s family history, including genealogy, national origins, places and dates of birth, etc.

*Sub-topic 1.* Find the TIME magazine photo essay “Barack Obama’s Family Tree.”

*Sub-topic 2.* Where did Barack Obama’s parents and grandparents come from?

*Sub-topic 3.* Find biographical information on Barack Obama’s mother.

This topic is rephrased as follows:

*Obama’s family.* Write about President Barack Obama’s family history, including genealogy, national origins, places and dates of birth, etc. Where did Barack Obama’s parents and grandparents come from? Also include a brief biography of Obama’s mother.

In the example, Sub-topic 1 is considered too specific for our purposes while the other sub-topics are retained. TREC Web Track topics divide into faceted and ambiguous topics. While topics of the first kind can be directly rephrased into essay topics, from topics of the second kind one of the available interpretations was chosen.

## 2.2 A Controlled Web Search Environment

To give the oDesk writers a familiar search experience while maintaining reproducibility at the same time, we developed a tailored search engine called ChatNoir (Potthast et al., 2012b).<sup>5</sup> Besides ours, the only other public search engine for the ClueWeb is Carnegie Mellon’s Indri,<sup>6</sup> which, unfortunately, is far from our efficiency requirements. Moreover, its search interface does not follow the standard in terms of result page design, and it does not give access to interaction logs. Our search engine is on the order of milliseconds in terms of retrieval

<sup>5</sup><http://chatnoir.webis.de>

<sup>6</sup><http://lemurproject.org/clueweb09.php/index.php#Services>

time, its interface follows industry standards, and it features an API that allows for user tracking.

ChatNoir is based on the BM25F retrieval model (Robertson et al., 2004), uses the anchor text list provided by (Hiemstra and Hauff, 2010), the PageRanks provided by the Carnegie Mellon University alongside the ClueWeb corpus, and the Spam rank list provided by (Cormack et al., 2011). ChatNoir comes with a proximity feature with variable-width buckets as described by (Elsayed et al., 2011). Our choice of retrieval model and ranking features is intended to provide a reasonable baseline performance. However, it is neither near as mature as those of commercial search engines nor does it compete with the best-performing models from TREC. Yet, it is among the most widely accepted models in information retrieval, which underlines our goal of reproducibility.

In addition to its retrieval model, ChatNoir implements two search facets: text readability scoring and long text search. The first facet, similar to that provided by Google, scores the readability of a text found on a web page via the well-known Flesch-Kincaid grade level formula (Kincaid et al., 1975): it estimates the number of years of education required in order to understand a given text. This number is mapped onto the three categories “Simple” (up to 5 years), “Intermediate” (between 5 and 9 years) and “Expert” (at least 9 years). The “Long Text” search facet omits search results which do not contain at least one continuous paragraph of text that exceeds 300 words. The two facets can be combined with each other.

When clicking on a search result, ChatNoir does not link into the real web but redirects into the ClueWeb. Though the ClueWeb provides the original URLs from which the web pages have been obtained, many of these pages have gone or been updated since. We hence set up an API that serves web pages from the ClueWeb on demand: when accessing a web page, it is pre-processed before being shipped, removing automatic referrers and replacing all links to the real web with links to their counterpart inside the ClueWeb. This way, the ClueWeb can be browsed as if surfing the real web, whereas it becomes possible to track a user. The ClueWeb is stored in the HDFS of our 40 node Hadoop cluster, and web pages are fetched directly from there with latencies of about 200ms. ChatNoir’s inverted index has been optimized to guarantee fast response times, and it is deployed alongside Hadoop on the same cluster.

Table 1: Demographics of the 12 Batch 2 writers.

Writer Demographics					
<i>Age</i>		<i>Gender</i>		<i>Native language(s)</i>	
Minimum	24	Female	67%	English	67%
Median	37	Male	33%	Filipino	25%
Maximum	65			Hindi	17%
<i>Academic degree</i>		<i>Country of origin</i>		<i>Second language(s)</i>	
Postgraduate	41%	UK	25%	English	33%
Undergraduate	25%	Philippines	25%	French	17%
None	17%	USA	17%	Afrikaans, Dutch,	
n/a	17%	India	17%	German, Spanish,	
		Australia	8%	Swedish each	8%
		South Africa	8%	None	8%
<i>Years of writing</i>		<i>Search engines used</i>		<i>Search frequency</i>	
Minimum	2	Google	92%	Daily	83%
Median	8	Bing	33%	Weekly	8%
Standard dev.	6	Yahoo	25%	n/a	8%
Maximum	20	Others	8%		

### 2.3 Two Batches of Writing

In order to not rely only on the retrieval model implemented in our controlled web search environment, we divided the task into two batches, so that two essays had to be written for each of the 150 topics, namely one in each batch. In Batch 1, our writers did not search for sources themselves, but they were provided up front with an average of 20 search results to choose from for each topic. These results were obtained from the TREC Web Track relevance judgments (so-called “qrels”): only documents that were found to be relevant or key documents for a given topic by manual inspection of the NIST assessors were provided to our writers. These documents result from the combined wisdom of all retrieval models of the TREC Web Tracks 2009–2011, and hence can be considered as optimum retrieval results produced by the state of the art in search engine technology. In Batch 2, in order to obtain realistic search interaction logs, our writers were instructed to search for source documents using ChatNoir.

### 2.4 Crowdsourcing Writers

Our ideal writer has experience in writing, is capable of writing about a diversity of topics, can complete a text in a timely manner, possesses decent English writing skills, and is well-versed in using the aforementioned technologies. After bootstrapping our setup with 10 volunteers recruited at our university, it became clear that, because of the workload involved, accomplishing our goals would not be possible with volunteers only. Therefore, we resorted to hiring (semi-)professional writers and made use of the crowdsourcing platform oDesk.<sup>7</sup> Crowdsourcing has quickly become one of the

<sup>7</sup><http://www.odesk.com>

Table 2: Key figures of the Webis text reuse corpus 2012.

Corpus characteristic	Distribution				Total
	min	avg	max	stdev	
Writers	<i>(Batch 1+2)</i>				27
Essays (Topics)	<i>(Two essays per topic)</i>				297 (150)
Essays / Writer	1	2	66	15.9	
Queries	<i>(Batch 2)</i>				13 655
Queries / Essay	4	91.0	616	83.1	
Clicks	<i>(Batch 2)</i>				16 739
Clicks / Essay	12	111.6	443	80.3	
Clicks / Query	1	2.3	76	3.3	
Irrelevant	<i>(Batch 2)</i>				5 962
Irrelevant / Essay	1	39.8	182	28.7	
Irrelevant / Query	0	0.5	60	1.4	
Relevant	<i>(Batch 2)</i>				251
Relevant / Essay	0	1.7	7	1.5	
Relevant / Query	0	0.0	4	0.2	
Key	<i>(Batch 2)</i>				1 937
Key / Essay	1	12.9	46	7.5	
Key / Query	0	0.2	22	0.7	

Corpus characteristic	Distribution				Total
	min	avg	max	stdev	
Search Sessions	<i>(Batch 2)</i>				931
Sessions / Essay	1	12.3	149	18.9	
Days	<i>(Batch 2)</i>				201
Days / Essay	1	4.9	17	2.7	
Hours	<i>(Batch 2)</i>				2 068
Hours / Writer	3	129.3	679	167.3	
Hours / Essay	3	7.5	10	2.5	
Edits	<i>(Batch 1+2)</i>				633 334
Edits / Essay	45	2 132.4	6 975	1 444.9	
Edits / Day	5	2 959.5	8 653	1 762.5	
Words	<i>(Batch 1+2)</i>				1 704 354
Words / Essay	260	5 738.8	15 851	1 604.3	
Words / Writer	2 078	63 124.2	373 975	89 246.7	
Sources	<i>(Batch 1+2)</i>				4 582
Sources / Essay	0	15.4	69	10.0	
Sources / Writer	5	169.7	1 065	269.6	

cornerstones for constructing evaluation corpora, which is especially true for paid crowdsourcing. Compared to Amazon’s Mechanical Turk (Barr and Cabrera, 2006), which is used more frequently than oDesk, there are virtually no workers at oDesk submitting fake results because of its advanced rating features for workers and employers. Moreover, oDesk tracks their workers by randomly taking screenshots, which are provided to employers in order to check whether the hours logged correspond to work-related activity. This allowed us to check whether our writers used our environment instead of other search engines and editors.

During Batch 2, we have conducted a survey among the twelve writers who worked for us at that time. Table 1 gives an overview of the demographics of these writers, based on a questionnaire and their resumes at oDesk. Most of them come from an English-speaking country, and almost all of them speak more than one language, which suggests a reasonably good education. Two thirds of the writers are female, and all of them have years of writing experience. Hourly wages were negotiated individually and range from 3 to 34 US dollars (dependent on skill and country of residence), with an average of about 12 US dollars. For ethical reasons, we payed at least the minimum wage of the respective countries involved. In total, we spent 20 468 US dollars to pay the writers—an amount that may be considered large compared to other scientific crowdsourcing efforts from the literature, but small in terms of the potential of crowdsourcing to make a difference in empirical science.

### 3 Corpus Analysis

This section presents selected results of a preliminary corpus analysis. We overview the data and shed some light onto the search and writing behavior of writers.

#### 3.1 Corpus Statistics

Table 2 shows key figures of the collected interaction logs, including the absolute numbers of queries, relevance judgments, working times, number of edits, words, and retrieved sources, as well as their relation to essays, writers, and work time, where applicable. On average, each writer wrote 2 essays while the standard deviation is 15.9, since one very prolific writer managed to write 66 essays.

From a total of 13 655 queries submitted by the writers within Batch 2, each essay got an average of 91 queries. The average number of results clicked per query is 2.3. For comparison, we computed the average number of clicks per query in the AOL query log (Pass et al., 2006), which is 2.0. In this regard, the behavior of our writers on individual queries does not differ much from that of the average AOL user in 2006. Most of the clicks that we recorded are search result clicks, whereas 2 457 of them are browsing clicks on web page links. Among the browsing clicks, 11.3% are clicks on links that point to the same web page (i.e., anchor links using the hash part of a URL). The longest click trail contains 51 unique web pages, but most trails are very short. This is a surprising result, since we expected a larger proportion of browsing clicks, but it also shows that our writers

relied heavily on the ChatNoir’s ranking. Regarding search facets, we observed that our writers used them only for about 7% of their queries. In these cases, the writers used either the “Long Text” facet, which retrieves web pages containing at least one continuous passage of at least 300 words, or set the desired reading level to “Expert.”

The query log of each writer in Batch 2 divides into 931 search sessions with an average of 12.3 sessions per topic. Here, a session is defined as a sequence of queries recorded for a given topic which is not divided by a break longer than 30 minutes. Despite other claims in the literature (Jones and Klinkner, 2008; Hagen et al., 2013) we argue that, in our case, sessions can be reliably identified by timeouts because we have a priori knowledge about which query belongs to which essay. Typically, completing an essay took 4.9 days, which includes to a long-lasting exploration of the topic at hand.

The 297 essays submitted within the two batches were written with a total of 633 334 edits. Each topic was edited 2 132 times on average, whereas the standard deviation gives an idea about how diverse the modifications of the reused text were. Writers were not specifically instructed to modify a text as much as possible—rather they were encouraged to paraphrase in order to foreclose the detection by an automatic text reuse detector. This way, our corpus captures each writer’s idea of the necessary modification effort to accomplish this goal. The average lengths of the essays is 5 739 words, but there are also some short essays if hardly any useful information could be found on the respective topics. About 15 sources have been reused in each essay, whereas some writers reused text from as many as 69 unique documents.

### 3.2 Relevance Judgments

In the essays from Batch 2, writers reused texts from web pages they found during their search. This forms an interesting relevance signal which allows us to separate web pages relevant to a given topic from those which are irrelevant. Following the terminology of TREC, we consider web pages from which text is reused as *key documents* for the respective essay’s topic, while web pages that are on a click trail leading to a key document are termed *relevant*. The unusually high number of key documents compared to relevant documents is explained by the fact that there are only few click trails of this kind, whereas most web pages

Table 3: Confusion matrix of TREC judgments versus writer judgments.

TREC judgment	Writer judgment			
	irrelevant	relevant	key	unjudged
spam (-2)	3	0	1	2 446
spam (-1)	64	4	18	16 657
irrelevant (0)	219	13	73	33 567
relevant (1)	114	8	91	10 676
relevant (2)	44	5	56	3 711
key (3)	12	0	8	526
unjudged	5 506	221	1 690	–

have been retrieved directly. The remainder of web pages that were viewed but discarded by our writers are considered as irrelevant.

Each year, the NIST assessors employed for the TREC conference manually review hundreds of web pages that have been retrieved by experimental retrieval systems that are submitted to the various TREC tracks. This was also the case for the TREC Web Tracks from which the topics of our corpus are derived. We have compared the relevance judgments provided by TREC for these tracks with the implicit judgments from our writers. Table 3 contrasts the two judgment scales in the form of a confusion matrix. TREC uses a six-point Likert scale ranging from -2 (extreme Spam) to 3 (key document). For 733 of the documents visited by our writers, TREC relevance judgments can be found. From these, 456 documents (62%) have been considered irrelevant for the purposes of reuse by our writers, however, the TREC assessor disagree with this judgment in 170 cases. Regarding the documents considered as key documents for reuse by our writers, the TREC assessors disagree on 92 of the 247 documents. An explanation for the disagreement can be found in the differences between the TREC ad hoc search task and our text reuse task: the information nuggets (small chunks of text) that satisfy specific factual information needs from the original TREC topics are not the same as the information “ingots” (big chunks of text) that satisfy our writers’ needs.

### 3.3 Research Behavior

To analyze the writers’ search behavior during essay writing in Batch 2, we have recorded detailed search logs of their queries while they used our search engine. Figure 2 shows for each of the 150 essays of this batch a curve of the percentage of queries at times between a writer’s first query and an essay’s completion. We have normalized the time axis and excluded working breaks of more

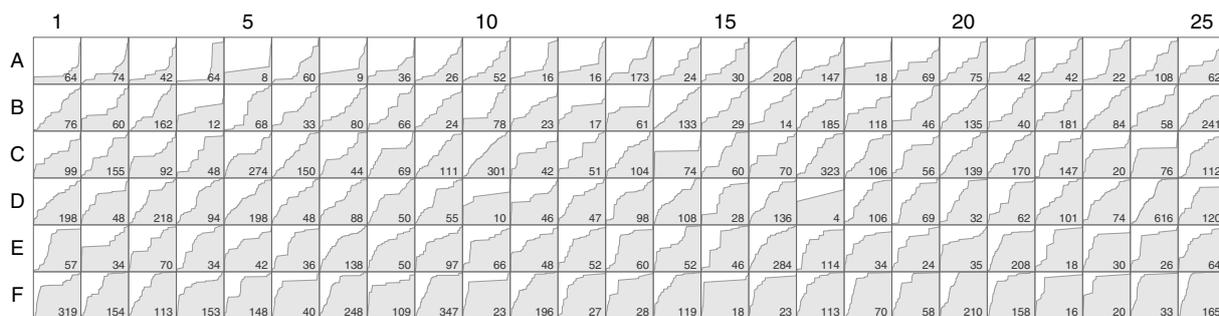


Figure 2: Spectrum of writer search behavior. Each grid cell corresponds to one of the 150 essays of Batch 2 and shows a curve of the percentage of submitted queries (y-axis) at times between the first query until the essay was finished (x-axis). The numbers denote the amount of queries submitted. The cells are sorted by area under the curve, from the smallest area in cell A1 to the largest area in cell F25.

than five minutes. The curves are organized so as to highlight the spectrum of different search behaviors we have observed: in row A, 70-90% of the queries are submitted toward the end of the writing task, whereas in row F almost all queries are submitted at the beginning. In between, however, sets of queries are often submitted in the form of “bursts,” followed by extended periods of writing, which can be inferred from the steps in the curves (e.g., cell C12). Only in some cases (e.g., cell C10) a linear increase of queries over time can be observed for a non-trivial amount of queries, which indicates continuous switching between searching and writing. From these observations, it can be inferred that our writers sometimes conducted a “first fit” search and reused the first texts they found easily. However, as the essay progressed and the low hanging fruit in terms of search were used up, they had to search more intensively in order to complete their essay. More generally, this data gives an idea of how humans perform exploratory search in order to learn about a given topic. Our current research on this aspect focuses on the prediction of search mission types, since we observe that the search mission type does not simply depend on the writer or the perceived topic difficulty.

### 3.4 Visualizing Edit Histories

To analyze the writers’ writing style, that is to say, how writers reuse texts and how the essay is completed in both batches, we have recorded the edit logs of their essays. Whenever a writer stopped writing for more than 300ms, a new edit was stored in a version control system at our site. The edit logs document the entire text evolution, from first the keystroke until an essay was completed. We have used the so-called history flow visualization to analyze the writing process (Vié-

gas et al., 2004). Figure 3 shows four examples from the set of 297 essays. Based on these visualizations, a number of observations can be made. In general, we identify two distinct writing-style types to perform text reuse, namely to *build up* an essay during writing, or, to first gather material and then to *boil down* a text until the essay is completed. Later in this section, we will analyze this observation in greater detail. Within the plots, a number of events can be spotted that occurred during writing: in the top left plot, encircled as area A, the insertion of a new piece of text can be observed. Though marked as original text at first, the writer worked on this passage and then revealed that it was reused from another source. At area B in the top right plot, one can observe the reorganization of two passages as they exchange places from one edit to another. Area C in the bottom right plot shows that the writer, shortly before completing this essay, reorganized substantial parts. Area D in the same plot shows how the writer went about boiling down the text by incorporating contents from different passages that have been collected beforehand and, then, from one edit to another, discarded most of the rest. The saw-tooth shaped pattern in area E in the bottom left plot reveals that, even though the writer of this essay adopts a build-up style, she still pastes passages from her sources into the text one at a time, and then individually boils down each. Our visualizations also include information about the text positions where writers have been working at a given point in time; these positions are shown as blue dots in the plots. In this regard distinct writing patterns are discernible of writers who go through a text linearly versus those who do not. Future work will include an analysis of these writing patterns.

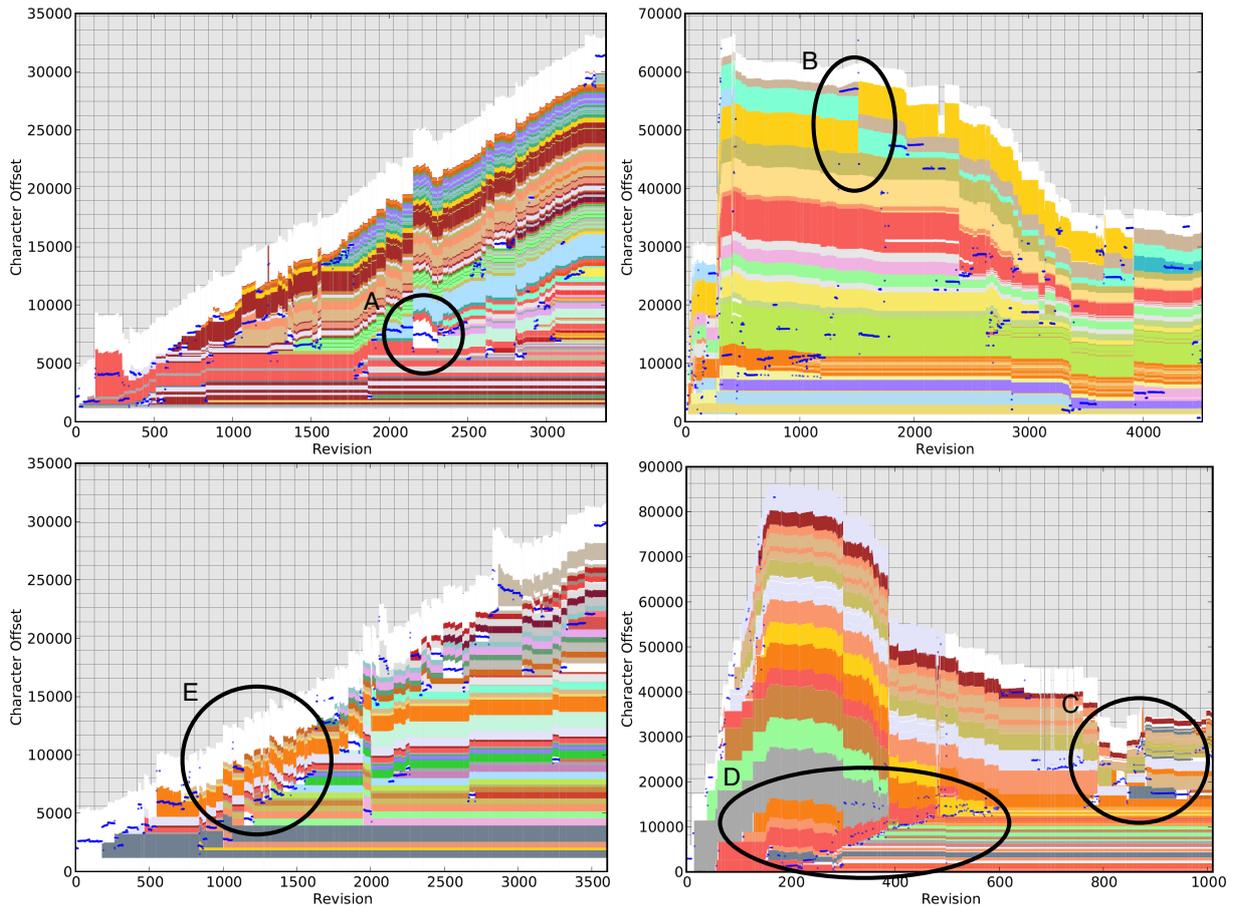


Figure 3: Types of text reuse: build-up reuse (left) versus boil-down reuse (right). Each plot shows the text length at text edit between first keystroke and essay completion; edits have been recorded during writing whenever a writer stopped for more than 300ms. Colors encode different source documents. Original text is white; blue dots indicate the text position of the writer’s last edit.

### 3.5 Build-up Reuse versus Boil-down Reuse

Based on the edit history visualizations, we have manually classified the 297 essays of both batches into two categories, corresponding to the two styles build-up reuse and boil-down reuse. We found that 40% are instances of build-up reuse, 45% are instances of boil-down reuse, and 13% fall in between, excluding 2% of the essays as outliers due to errors or for being too short. The in-between cases show that a writer actually started one way and then switched to the respective other style of reuse so that the resulting essays could not be attributed to a single category. An important question that arises out of this observation is whether different writers habitually exert different reuse styles or whether they apply them at random. To obtain a better overview, we envision the applied reuse style of an essay by the skyline curve of its edit history visualization (i.e., by the curve that plots the length of an essay after each edit). Aggregating these curves on a per-writer basis reveals distinct

Table 4: Contingency table: writers over reuse style.

Reuse Style	Writer ID											
	A02	A05	A06	A07	A10	A17	A18	A19	A20	A21	A24	
build-up	4	27	11	4	9	13	12	4	9	18	2	
boil-down	52	5	0	14	2	13	11	3	0	0	24	
mixed	10	3	0	1	1	7	6	0	0	3	1	

patterns. For eight of our writers Figure 4 shows this characteristic. The plots are ordered by the shape of the averaged curve, starting from a linear increase (left) to a compound of steep increase to a certain length after which the curve levels out (right). The former shape corresponds to writers who typically apply build-up reuse, while the latter can be attributed to writers who typically apply boil-down reuse.

When comparing the plots we notice a very interesting effect: it appears that writers who conduct boil-down reuse vary more wildly in their behavior. The reuse style of some writers, however, falls in between the two extremes. Besides the visual analysis, Table 4 shows the distribution of reuse styles

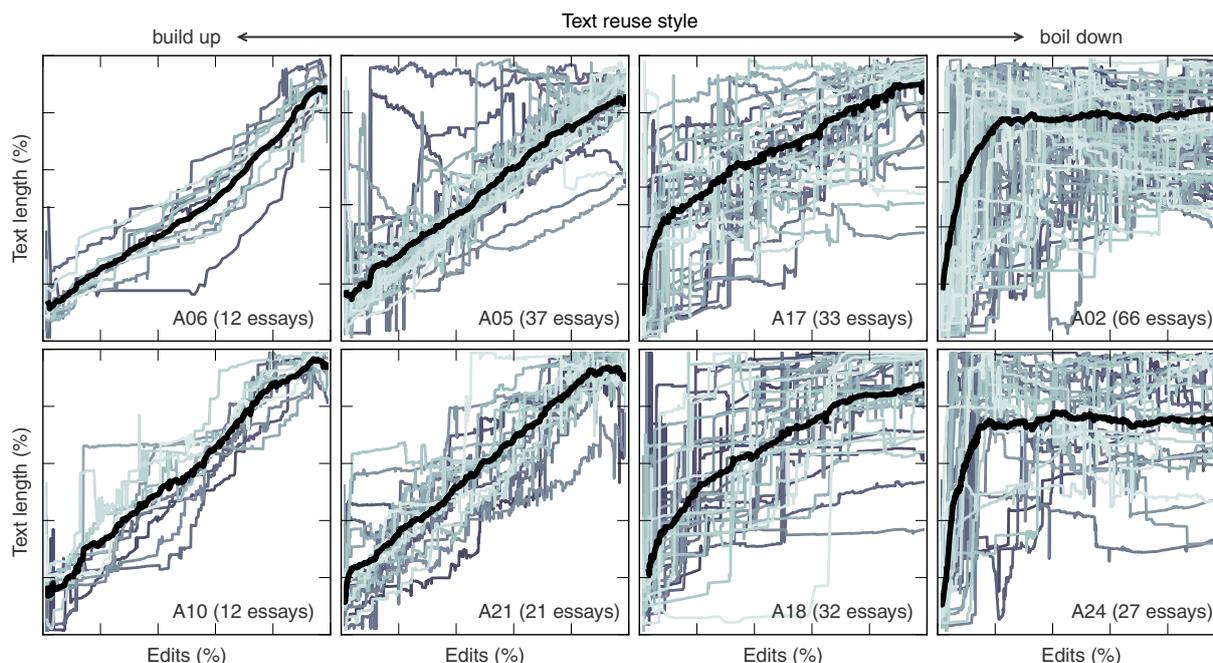


Figure 4: Text reuse styles ranging from build-up reuse (left) to boil-down reuse (right). A gray curve shows the normalized length of an essay over the edits that went into it during writing. Curves are grouped by writers. The black curve marks the average of all other curves in a plot.

for the eleven writers who contributed at least five essays. Most writers use one style for about 80% of their essays, whereas two writers (A17, A18) are exactly on par between the two styles. Based on Pearson’s chi-squared test, one can safely reject the null hypothesis that writers and text reuse styles are independent:  $\chi^2 = 139.0$  with  $p = 7.8 \cdot 10^{-20}$ . Since our sample of authors and essays is sparse, Pearson’s chi-squared test may not be perfectly suited which is why we have also applied Fisher’s exact test, which computes probability  $p = 0.0005$  that the null hypothesis is true.

#### 4 Summary and Outlook

This paper details the construction of the Webis text reuse corpus 2012 (Webis-TRC-12), a new corpus for text reuse research that has been created entirely manually on a large scale. We have recorded consistent interaction logs of human writers with a search engine as well as with the used text processor; these logs serve the purpose of studying how texts from the web are being reused for essay writing. Our setup is entirely reproducible: we have built a static web search environment consisting of a search engine along with a means to browse a large corpus of web pages as if it were the “real” web. Yet, in terms of scale, this environment is representative of the real web. Besides our corpus also this infrastructure is available to other researchers.

The corpus itself goes beyond existing resources in that it allows for a much more fine-grained analysis of text reuse, and in that it significantly improves the realism of the data underlying evaluations of automatic tools to detect text reuse and plagiarism.

Our analysis gives an overview of selected aspects of the new corpus. This includes corpus statistics about important variables, but also exploratory studies of search behaviors and strategies for reusing text. We present new insights about how text is composed, revealing two types of writers: those who build up a text as they go, and those who first collect a lot of material which then is boiled down until the essay is finished.

Parts of our corpus have been successfully employed to evaluate plagiarism detectors in the PAN plagiarism detection competition 2012 (Potthast et al., 2012a). Future work will include analyses that may help to understand the state of mind of writers when reusing text as well as of plagiarists. We also expect insights with regard to the development of algorithms for detection purposes and for linguists studying the process of writing.

#### Acknowledgements

We thank our writers at oDesk and all volunteers for their contribution. We also thank Jan Graßegger and Martin Tippmann who kept the search engine up and running during corpus construction.

## References

- Jeff Barr and Luis Felipe Cabrera. 2006. AI gets a brain. *Queue*, 4(4):24–29.
- Paul Clough and Mark Stevenson. 2011. Developing a corpus of plagiarised short answers. *Language Resources and Evaluation*, 45:5–24.
- Paul Clough, Robert Gaizauskas, Scott S. L. Piao, and Yorick Wilks. 2002. METER: MEasuring TExt Reuse. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, Philadelphia, PA, USA, July 6–12, 2002, pages 152–159.
- Gordon V. Cormack, Mark D. Smucker, and Charles L. A. Clarke. 2011. Efficient and effective spam filtering and re-ranking for large web datasets. *Information Retrieval*, 14(5):441–465.
- Tamer Elsayed, Jimmy J. Lin, and Donald Metzler. 2011. When close enough is good enough: approximate positional indexes for efficient ranked retrieval. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM 2011)*, Glasgow, United Kingdom, October 24–28, 2011, pages 1993–1996.
- Matthias Hagen, Jakob Gomoll, Anna Beyer, and Benno Stein. 2013. From Search Session Detection to Search Mission Detection. In *Proceedings of the 10th International Conference Open Research Areas in Information Retrieval (OAIR 2013)*, Lisbon, Portugal, May 22–24, 2013, to appear.
- Djoerd Hiemstra and Claudia Hauff. 2010. MIREX: MapReduce information retrieval experiments. Technical Report TR-CTIT-10-15, University of Twente.
- Rosie Jones and Kristina Lisa Klinkner. 2008. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM 2008)*, Napa Valley, California, USA, October 26–30, 2008, pages 699–708.
- J. Peter Kincaid, Robert P. Fishburne, Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, Fog count and Flesch reading ease formula) for Navy enlisted personnel. Research Branch Report 8-75, Naval Air Station Memphis, Millington, TN.
- Katrin Köhler and Debora Weber-Wulff. 2010. Plagiarism detection test 2010. <http://plagiat.htw-berlin.de/wp-content/uploads/PlagiarismDetectionTest2010-final.pdf>.
- Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems (Infoscale 2006)*, Hong Kong, May 30–June 1, 2006, paper 1.
- Martin Potthast, Benno Stein, Andreas Eiselt, Alberto Barrón-Cedeño, and Paolo Rosso. 2009. Overview of the 1st international competition on plagiarism detection. In *SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 2009)*, pages 1–9.
- Martin Potthast, Alberto Barrón-Cedeño, Andreas Eiselt, Benno Stein, and Paolo Rosso. 2010a. Overview of the 2nd international competition on plagiarism detection. In *Working Notes Papers of the CLEF 2010 Evaluation Labs*.
- Martin Potthast, Benno Stein, Alberto Barrón-Cedeño, and Paolo Rosso. 2010b. An evaluation framework for plagiarism detection. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, Beijing, China, August 23–27, 2010, pages 997–1005.
- Martin Potthast, Andreas Eiselt, Alberto Barrón-Cedeño, Benno Stein, and Paolo Rosso. 2011. Overview of the 3rd international competition on plagiarism detection. In *Working Notes Papers of the CLEF 2011 Evaluation Labs*.
- Martin Potthast, Tim Gollub, Matthias Hagen, Jan Graßegger, Johannes Kiesel, Maximilian Michel, Arnd Oberländer, Martin Tippmann, Alberto Barrón-Cedeño, Parth Gupta, Paolo Rosso, and Benno Stein. 2012a. Overview of the 4th international competition on plagiarism detection. In *Working Notes Papers of the CLEF 2012 Evaluation Labs*.
- Martin Potthast, Matthias Hagen, Benno Stein, Jan Graßegger, Maximilian Michel, Martin Tippmann, and Clement Welsch. 2012b. ChatNoir: a search engine for the ClueWeb09 corpus. In *Proceedings of the 35th International ACM Conference on Research and Development in Information Retrieval (SIGIR 2012)*, Portland, OR, USA, August 12–16, 2012, page 1004.
- Martin Potthast. 2011. *Technologies for Reusing Text from the Web*. Dissertation, Bauhaus-Universität Weimar.
- Stephen E. Robertson, Hugo Zaragoza, and Michael J. Taylor. 2004. Simple BM25 extension to multiple weighted fields. In *Proceedings of the 13th ACM Conference on Information and Knowledge Management (CIKM 2004)*, Washington, DC, USA, November 8–13, 2004, pages 42–49.
- Fernanda B. Viégas, Martin Wattenberg, and Kushal Dave. 2004. Studying cooperation and conflict between authors with history flow visualizations. In *Proceedings of the 2004 Conference on Human Factors in Computing Systems (CHI 2004)*, Vienna, Austria, April 24–29, 2004, pages 575–582.

# SPred: Large-scale Harvesting of Semantic Predicates

Tiziano Flati and Roberto Navigli

Dipartimento di Informatica

Sapienza Università di Roma

{flati,navigli}@di.uniroma1.it

## Abstract

We present SPred, a novel method for the creation of large repositories of semantic predicates. We start from existing collocations to form lexical predicates (e.g., *break* \*) and learn the semantic classes that best fit the \* argument. To do this, we extract all the occurrences in Wikipedia which match the predicate and abstract its arguments to general semantic classes (e.g., *break* BODY PART, *break* AGREEMENT, etc.). Our experiments show that we are able to create a large collection of semantic predicates from the Oxford Advanced Learner's Dictionary with high precision and recall, and perform well against the most similar approach.

## 1 Introduction

Acquiring semantic knowledge from text automatically is a long-standing issue in Computational Linguistics and Artificial Intelligence. Over the last decade or so the enormous abundance of information and data that has become available has made it possible to extract huge amounts of patterns and named entities (Etzioni et al., 2005), semantic lexicons for categories of interest (Thelen and Riloff, 2002; Igo and Riloff, 2009), large domain glossaries (De Benedictis et al., 2013) and lists of concepts (Katz et al., 2003). Recently, the availability of Wikipedia and other collaborative resources has considerably boosted research on several aspects of knowledge acquisition (Hovy et al., 2013), leading to the creation of several large-scale knowledge resources, such as DBpedia (Bizer et al., 2009), BabelNet (Navigli and Ponzetto, 2012), YAGO (Hoffart et al., 2013), MENTA (de Melo and Weikum, 2010), to name but a few. This wealth of acquired knowledge is known to have a positive impact on important fields such as Information Retrieval (Chu-Carroll and Prager, 2007), Information Extraction (Krause

et al., 2012), Question Answering (Ferrucci et al., 2010) and Textual Entailment (Berant et al., 2012; Stern and Dagan, 2012).

Not only are these knowledge resources obtained by acquiring concepts and named entities, but they also provide semantic relations between them. These relations are extracted from unstructured or semi-structured text using ontology learning from scratch (Velardi et al., 2013) and Open Information Extraction techniques (Etzioni et al., 2005; Yates et al., 2007; Wu and Weld, 2010; Fader et al., 2011; Moro and Navigli, 2013) which mainly stem from seminal work on *is-a* relation acquisition (Hearst, 1992) and subsequent developments (Girju et al., 2003; Pasca, 2004; Snow et al., 2004, among others).

However, these knowledge resources still lack semantic information about language units such as phrases and collocations. For instance, which semantic classes are expected as a direct object of the verb *break*? What kinds of noun does the adjective *amazing* collocate with? Recognition of the need for systems that are aware of the selectional restrictions of verbs and, more in general, of textual expressions, dates back to several decades (Wilks, 1975), but today it is more relevant than ever, as is testified by the current interest in semantic class learning (Kozareva et al., 2008) and supertype acquisition (Kozareva and Hovy, 2010). These approaches leverage lexico-syntactic patterns and input seeds to recursively learn the semantic classes of relation arguments. However, they require the manual selection of one or more seeds for each pattern of interest, and this selection influences the amount and kind of semantic classes to be learned. Furthermore, the learned classes are not directly linked to existing resources such as WordNet (Fellbaum, 1998) or Wikipedia.

The goal of our research is to create a large-scale repository of semantic predicates whose lexical arguments are replaced by their semantic classes. For example, given the textual expression *break a toe* we want to create the correspond-

ing semantic predicate *break a BODY PART*, where BODY PART is a class comprising several lexical realizations, such as *leg, arm, foot*, etc.

This paper provides three main contributions:

- We propose SPred, a novel approach which harvests predicates from Wikipedia and generalizes them by leveraging core concepts from WordNet.
- We create a large-scale resource made up of semantic predicates.
- We demonstrate the high quality of our semantic predicates, as well as the generality of our approach, also in comparison with our closest competitor.

## 2 Preliminaries

We introduce two preliminary definitions which we use in our approach.

**Definition 1 (lexical predicate).** A lexical predicate  $w_1 w_2 \dots w_i * w_{i+1} \dots w_n$  is a regular expression, where  $w_j$  are tokens ( $j = 1, \dots, n$ ),  $*$  matches any sequence of one or more tokens, and  $i \in \{0, \dots, n\}$ . We call the token sequence which matches  $*$  the filling argument of the predicate.

For example,  $a * \textit{of milk}$  matches occurrences such as *a full bottle of milk, a glass of milk, a carton of milk*, etc. While in principle  $*$  could match any sequence of words, since we aim at generalizing nouns, in what follows we allow  $*$  to match only noun phrases (e.g., *glass, hot cup, very big bottle*, etc.).

**Definition 2 (semantic predicate).** A semantic predicate is a sequence  $w_1 w_2 \dots w_i c w_{i+1} \dots w_n$ , where  $w_j$  are tokens ( $j = 1, \dots, n$ ),  $c \in C$  is a semantic class selected from a fixed set  $C$  of classes, and  $i \in \{0, \dots, n\}$ .

As an example, consider the semantic predicate *cup of BEVERAGE*,<sup>1</sup> where BEVERAGE is a semantic class representing beverages. This predicate matches phrases like *cup of coffee, cup of tea*, etc., but not *cup of sky*. Other examples include: *MUSICAL INSTRUMENT is played by*, *a CONTAINER of milk, break AGREEMENT*, etc.

Semantic predicates mix the lexical information of a given lexical predicate with the explicit semantic modeling of its argument. Importantly, the same lexical predicate can have different classes as its argument, like *cup of FOOD* vs. *cup of BEVERAGE*. Note, however, that different classes might convey different semantics for the same lexical

predicate, such as *cup of COUNTRY*, referring to cup as a prize instead of cup as a container.

## 3 Large-Scale Harvesting of Semantic Predicates

The goal of this paper is to provide a fully automatic approach for the creation of a large repository of semantic predicates in three phases. For each lexical predicate of interest (e.g., *break \**):

1. We extract all its possible filling arguments from Wikipedia, e.g., *lease, contract, leg, arm*, etc. (Section 3.1).
2. We disambiguate as many filling arguments as possible using Wikipedia, obtaining a set of corresponding Wikipedia pages, e.g., *Lease, Contract*, etc. (Section 3.2).
3. We create the semantic predicates by generalizing the Wikipedia pages to their most suitable semantic classes, e.g., *break AGREEMENT, break LIMB*, etc. (Section 3.3).

We can then exploit the learned semantic predicates to assign the most suitable semantic class to new filling arguments for the given lexical predicate (Section 3.4).

### 3.1 Extraction of Filling Arguments

Let  $\pi$  be an input lexical predicate (e.g., *break \**). We search the English Wikipedia for all the token sequences which match  $\pi$ , resulting in a list of noun phrases filling the  $*$  argument. We show an excerpt of the output obtained when searching Wikipedia for the arguments of the lexical predicate  $a * \textit{of milk}$  in Table 1. As can be seen, a wide range of noun phrases are extracted, from quantities such as *glass* and *cup* to other aspects, such as *brand* and *constituent*.

The output of this first step is a set  $L_\pi$  of triples  $(a, s, l)$  of filling arguments  $a$  matching the lexical predicate  $\pi$  in a sentence  $s$  of the Wikipedia corpus, with  $a$  potentially linked to a page  $l$  (e.g., see the top 3 rows in Table 1;  $l = \epsilon$  if no link is provided, see bottom rows of the Table).<sup>2</sup> Note that Wikipedia is the only possible corpus that can be used here for at least two reasons: first, in order to extract relevant arguments, we need a large corpus of a definitional nature; second, we need wide-coverage semantic annotations of filling arguments.

### 3.2 Disambiguation of Filling Arguments

The objective of the second step is to disambiguate as many arguments in  $L_\pi$  as possible for the lex-

<sup>1</sup>In what follows we denote the SEMANTIC CLASS in small capitals and the *lexical predicate* in italics.

<sup>2</sup>We will also refer to  $l$  as the *sense* of  $a$  in sentence  $s$ .

a	full [[bottle]]	of milk
a	nice hot [[cup]]	of milk
a	cold [[glass]]	of milk
a	very big bottle	of milk
a	brand	of milk
a	constituent	of milk

Table 1: An excerpt of the token sequences which match the lexical predicate  $a * of\ milk$  in Wikipedia (filling argument shown in the second column; following the Wikipedia convention we provide links in double square brackets).

ical predicate  $\pi$ . We denote  $D_\pi = \{(a, s, l) : l \neq \epsilon\} \subseteq L_\pi$  as the set of those arguments originally linked to the corresponding Wikipedia page (like the top three linked arguments in Table 1). Therefore, in the rest of this section we will focus only on the remaining triples  $(a, s, \epsilon) \in U_\pi$ , where  $U_\pi = L_\pi \setminus D_\pi$ , i.e., those triples whose arguments are not semantically annotated. Our goal is to replace  $\epsilon$  with an appropriate sense, i.e., page, for  $a$ . For each such triple  $(a, s, \epsilon) \in U_\pi$ , we apply the following disambiguation heuristics:

- **One sense per page:** if another occurrence of  $a$  in the same Wikipedia page (independent of the lexical predicate) is linked to a page  $l$ , then remove  $(a, s, \epsilon)$  from  $U_\pi$  and add  $(a, s, l)$  to  $D_\pi$ . In other words, we propagate an existing annotation of  $a$  in the same Wikipedia page and apply it to our ambiguous item. For instance, *cup of coffee* appears in the Wikipedia page *Energy drink* in the sentence “[...] energy drinks contain more caffeine than a strong *cup of coffee*”, but this occurrence of *coffee* is not linked. However the second paragraph contains the sentence “[[Coffee]], tea and other naturally caffeinated beverages are usually not considered energy drinks”, where *coffee* is linked to the *Coffee* page. This heuristic naturally reflects the broadly known assumption about lexical ambiguity presented in (Yarowsky, 1995), namely the one-sense-per-discourse heuristic.
- **One sense per lexical predicate:** if  $\exists(a, s', l) \in D_\pi$ , then remove  $(a, s, \epsilon)$  from  $U_\pi$  and add  $(a, s, l)$  to  $D_\pi$ . If multiple senses of  $a$  are available, choose the most frequent one in  $D_\pi$ . For example, in the page *Singaporean cuisine* the occurrence of *coffee* in the sentence “[...] combined with a *cup of coffee* and a half-boiled egg” is not linked, but we have collected many other occurrences, all linked to the *Coffee* page, so this link

gets propagated to our ambiguous item as well. This heuristic mimes the one-sense-per-collocation heuristic presented in (Yarowsky, 1995).

- **Trust the inventory:** if Wikipedia provides only one sense for  $a$ , i.e., only one page title whose lemma is  $a$ , link  $a$  to that page. Consider the instance “At that point, Smith threw down a *cup of Gatorade*” in page *Jimmy Clausen*; there is only one sense for *Gatorade* in Wikipedia, so we link the unannotated occurrence to it.

As a result, the initial set of disambiguated arguments in  $D_\pi$  is augmented with all those triples for which any of the above three heuristics apply. Note that  $D_\pi$  might contain the same argument several times, occurring in different sentences and linked many times to the same page or to different pages. Notably, the discovery of new links is made through one scan of Wikipedia per heuristic. The three disambiguation strategies, applied in the same order as presented above, contribute to promoting the most relevant sense for a given word.

Finally, let  $A$  be the set of arguments in  $D_\pi$ , i.e.,  $A := \{a : \exists(a, s, l) \in D_\pi\}$ . For each argument  $a \in A$  we select the majority sense  $sense(a)$  of  $a$  and collect the corresponding set of sentences  $sent(a)$  marked with that sense. Formally,  $sense(a) := \arg \max_l |\{(x, y, z) \in D_\pi : x = a \wedge z = l\}|$  and  $sent(a) := \{s : (a, s, sense(a)) \in D_\pi\}$ .

### 3.3 Generalization to Semantic Classes

Our final objective is to generalize the annotated arguments to semantic classes picked out from a fixed set  $C$  of classes. As explained below, we assume the set  $C$  to be made up of representative synsets from WordNet. We perform this in two substeps: we first link all our disambiguated arguments to WordNet (Section 3.3.1) and then leverage the WordNet taxonomy to populate the semantic classes in  $C$  (Section 3.3.2).

#### 3.3.1 Linking to WordNet

So far the arguments in  $D_\pi$  have been semantically annotated with the Wikipedia pages they refer to. However, using Wikipedia as our sense inventory is not desirable; in fact, contrarily to other commonly used lexical-semantic networks such as WordNet, Wikipedia is not formally organized in a structured, taxonomic hierarchy. While it is true that attached to each Wikipedia page there are one or more categories, these categories just provide shallow information about the class the page

belongs to. Indeed, categories are not ideal for representing the semantic classes of a Wikipedia page for at least three reasons: i) many categories do not express taxonomic information (e.g., the English page *Albert Einstein* provides categories such as DEATHS FROM ABDOMINAL AORTIC ANEURYSM and INSTITUTE FOR ADVANCED STUDY FACULTY); ii) categories are mostly structured in a directed acyclic graph with multiple parents per category (even worse, cycles are possible in principle); iii) there is no clear way of identifying core semantic classes from the large set of available categories. Although efforts towards the automatic taxonomization of Wikipedia categories do exist in the literature (Ponzetto and Strube, 2011; Nastase and Strube, 2013), the results are of a lower quality than a hand-built lexical resource. Therefore, as was done in previous work (Mihalcea and Moldovan, ; Ciaramita and Altun, 2006; Izquierdo et al., 2009; Erk and McCarthy, 2009; Huang and Riloff, 2010), we pick out our semantic classes  $C$  from WordNet and leverage its manually-curated taxonomy to associate our arguments with the most suitable class. This way we avoid building a new taxonomy and shift the problem to that of projecting the Wikipedia pages – associated with annotated filling arguments – to synsets in WordNet. We address this problem in two steps:

**Wikipedia-WordNet mapping.** We exploit an existing mapping implemented in BabelNet (Navigli and Ponzetto, 2012), a wide-coverage multilingual semantic network that integrates Wikipedia and WordNet.<sup>3</sup> Based on a disambiguation algorithm, BabelNet establishes a mapping  $\mu : \text{Wikipages} \rightarrow \text{Synsets}$  which links about 50,000 pages to their most suitable WordNet senses.<sup>4</sup>

**Mapping extension.** Nevertheless, BabelNet is able to solve the problem only partially, because it still leaves the vast majority of the 4 million English Wikipedia pages unmapped. This is mainly due to the encyclopedic nature of most pages, which do not have a counterpart in the WordNet dictionary. To address this issue, for each unmapped Wikipedia page  $p$  we obtain its textual definition as the first sentence of the page.<sup>5</sup> Next,

<sup>3</sup><http://babelnet.org>

<sup>4</sup>We follow (Navigli, 2009) and denote with  $w_p^i$  the  $i$ -th sense of  $w$  in WordNet with part of speech  $p$ .

<sup>5</sup>According to the Wikipedia guidelines, “The article should begin with a short declarative sentence, answering two questions for the nonspecialist reader: *What (or who) is the subject?* and *Why is this subject notable?*”, extracted from <http://en.wikipedia.org/wiki/>

we extract the hypernym from the textual definition of  $p$  by applying Word-Class Lattices (Navigli and Velardi, 2010, WCL<sup>6</sup>), a domain-independent hypernym extraction system successfully applied to taxonomy learning from scratch (Velardi et al., 2013) and freely available online (Faralli and Navigli, 2013). If a hypernym  $h$  is successfully extracted and  $h$  is linked to a Wikipedia page  $p'$  for which  $\mu(p')$  is defined, then we extend the mapping by setting  $\mu(p) := \mu(p')$ . For instance, the mapping provided by BabelNet does not provide any link for the page *Peter Spence*; thanks to WCL, though, we are able to set the page *Journalist* as its hypernym, and link it to the WordNet synset *journalist* <sub>$n$</sub> <sup>1</sup>.

This way our mapping extension now covers 539,954 pages, i.e., more than an order of magnitude greater than the number of pages originally covered by the BabelNet mapping.

### 3.3.2 Populating the Semantic Classes

We now proceed to populating the semantic classes in  $C$  with the annotated arguments obtained for the lexical predicate  $\pi$ .

**Definition 3 (semantic class of a synset).** The semantic class for a WordNet synset  $S$  is the class  $c$  among those in  $C$  which is the most specific hypernym of  $S$  according to the WordNet taxonomy.

For instance, given the synset *tap water* <sub>$n$</sub> <sup>1</sup>, its semantic class is *water* <sub>$n$</sub> <sup>1</sup> (while the other more general subsumers in  $C$  are not considered, e.g., *compound* <sub>$n$</sub> <sup>2</sup>, *chemical* <sub>$n$</sub> <sup>1</sup>, *liquid* <sub>$n$</sub> <sup>3</sup>, etc).

For each argument  $a \in A$  for which a Wikipedia-to-WordNet mapping  $\mu(\text{sense}(a))$  could be established as a result of the linking procedure described above, we associate  $a$  with the semantic class of  $\mu(\text{sense}(a))$ . For example, consider the case in which  $a$  is equal to *tap water* and  $\text{sense}(a)$  is equal to the Wikipedia page *Tap water*, in turn mapped to *tap water* <sub>$n$</sub> <sup>1</sup> via  $\mu$ ; we thus associate *tap water* with its semantic class *water* <sub>$n$</sub> <sup>1</sup>. If more than one class can be found we add  $a$  to each of them.<sup>7</sup>

Ultimately, for each class  $c \in C$ , we obtain a set  $\text{support}(c)$  made up of all the arguments  $a \in A$  associated with  $c$ . For instance,  $\text{support}(\text{beverage}_n^1) = \{ \text{chinese tea}, \text{3.2\% beer}, \text{hot cocoa}, \text{cider}, \dots, \text{orange juice} \}$ . Note that, thanks to our extended mapping (cf. Section 3.3.1), the support of a class can also contain arguments not covered in WordNet (e.g., *hot cocoa* and *tejuino*).

Wikipedia:Writing\_better\_articles.

<sup>6</sup><http://lcl.uniroma1.it/wcl>

<sup>7</sup>This can rarely happen due to multiple hypernyms available in WordNet for the same synset.

$P_{class}(c \pi)$	$c$	$support(c)$
0.1896	wine <sub>n</sub> <sup>1</sup>	wine, sack, white wine, red wine, wine in china, madeira wine, claret, kosher wine
0.1805	coffee <sub>n</sub> <sup>1</sup>	turkish coffee, drip coffee, espresso, coffee, cappucino, caffè latte, decaffeinated coffee, latte
0.1143	herb <sub>n</sub> <sup>2</sup>	green tea, indian tea, black tea, orange pekoe tea, tea
0.1104	water <sub>n</sub> <sup>1</sup>	water, seawater
0.0532	beverage <sub>n</sub> <sup>1</sup>	chinese tea, 3.2% beer, orange soda, boiled water, hot chocolate, hot cocoa, tejuino, cider, beverage, cocoa, coffee milk, lemonade, orange juice
0.0403	milk <sub>n</sub> <sup>1</sup>	skim milk, milk, cultured buttermilk, whole milk
0.0351	beer <sub>n</sub> <sup>1</sup>	3.2% beer, beer
0.0273	alcohol <sub>n</sub> <sup>1</sup>	mead, umeshu, kava, rice wine, jägermeister, kvass, sake, gin, rum
0.0182	poison <sub>n</sub> <sup>1</sup>	poison

Table 2: Highest-probability semantic classes for the lexical predicate  $\pi = \text{cup of } *$ , according to our set  $C$  of semantic classes.

Since not all classes are equally relevant to the lexical predicate  $\pi$ , we estimate the conditional probability of each class  $c \in C$  given  $\pi$  on the basis of the number of sentences which contain an argument in that class. Formally:

$$P_{class}(c|\pi) = \frac{\sum_{a \in support(c)} |sent(a)|}{Z}, \quad (1)$$

where  $Z$  is a normalization factor calculated as  $Z = \sum_{c' \in C} \sum_{a \in support(c')} |sent(a)|$ . As an example, in Table 2 we show the highest-probability classes for the lexical predicate *cup of*  $*$ .

As a result of the probabilistic association of each semantic class  $c$  with a target lexical predicate  $w_1 w_2 \dots w_i * w_{i+1} \dots w_n$ , we obtain a semantic predicate  $w_1 w_2 \dots w_i c w_{i+1} \dots w_n$ .

### 3.4 Classification of new arguments

Once the semantic predicates for the input lexical predicate  $\pi$  have been learned, we can classify a new filling argument  $a$  of  $\pi$ . However, the class probabilities calculated with Formula 1 might not provide reliable scores for several classes, including unseen ones whose probability would be 0.

To enable wide coverage we estimate a second conditional probability based on the distributional semantic profile of each class. To do this, we perform three steps:

1. For each WordNet synset  $S$  we create a distributional vector  $\vec{S}$  summing the noun occurrences within all the Wikipedia pages  $p$  such that  $\mu(p) = S$ . Next, we create a distributional vector for each class  $c \in C$  as follows:

$$\vec{c} = \sum_{S \in desc(c)} \vec{S},$$

where  $desc(c)$  is the set of all synsets which are descendants of the semantic class  $c$  in WordNet. As a result we obtain a predicate-independent distributional description for each semantic class in  $C$ .

2. Now, given an argument  $a$  of a lexical predicate  $\pi$ , we create a distributional vector  $\vec{a}$  by summing the noun occurrences of all the sentences  $s$  such that  $(a, s, l) \in L_\pi$  (cf. Section 3.1).
3. Let  $C_a$  be the set of candidate semantic classes for argument  $a$ , i.e.,  $C_a$  contains the semantic classes for the WordNet synsets of  $a$  as well as the semantic classes associated with  $\mu(p)$  for all Wikipedia pages  $p$  whose lemma is  $a$ . For each candidate class  $c \in C_a$ , we determine the cosine similarity between the distributional vectors  $\vec{c}$  and  $\vec{a}$  as follows:

$$sim(\vec{c}, \vec{a}) = \frac{\vec{c} \cdot \vec{a}}{\|\vec{c}\| \|\vec{a}\|}.$$

Then, we determine the most suitable semantic class  $c \in C_a$  of argument  $a$  as the class with the highest distributional probability, estimated as:

$$P_{distr}(c|\pi, a) = \frac{sim(\vec{c}, \vec{a})}{\sum_{c' \in C_a} sim(\vec{c}', \vec{a})}. \quad (2)$$

We can now choose the most suitable class  $c \in C_a$  for argument  $a$  which maximizes the probability mixture of the distributional probability in Formula 2 and the class probability in Formula 1:

$$P(c|\pi, a) = \alpha P_{distr}(c|\pi, a) + (1 - \alpha) P_{class}(c|\pi), \quad (3)$$

where  $\alpha \in [0, 1]$  is an interpolation factor.

We now illustrate the entire process of our algorithm on a real example. Given a textual expression such as *virus replicate*, we: (i) extract all the filling arguments of the lexical predicate  $* \text{replicate}$ ; (ii) link and disambiguate the extracted filling arguments; (iii) query our system for the available *virus* semantic classes (i.e.,  $\{virus_n^1, virus_n^3\}$ ); (iv) build the distributional vectors for

the candidate semantic classes and the given input argument; (v) calculate the probability mixture. As a result we obtain the following ranking,  $virus_n^1:0.250$ ,  $virus_n^3:0.000894$ , so that the first sense of *virus* in WordNet 3.0 is preferred, being an “ultramicroscopic infectious agent that replicates itself only within cells of living hosts”.

## 4 Experiment 1: Oxford Lexical Predicates

We evaluate on the two forms of output produced by SPred: (i) the top-ranking semantic classes of a lexical predicate, as obtained with Formula 1, and (ii) the classification of a lexical predicate’s argument with the most suitable semantic class, as produced using Formula 3. For both evaluations, we use a lexical predicate dataset built from the Oxford Advanced Learner’s Dictionary (Crowther, 1998).

### 4.1 Set of Semantic Classes

The selection of which semantic classes to include in the set  $C$  is of great importance. In fact, having too many classes will end up in an overly fine-grained inventory of meanings, whereas an excessively small number of classes will provide little discriminatory power. As our set  $C$  of semantic classes we selected the standard set of 3,299 core nominal synsets available in WordNet.<sup>8</sup> However, our approach is flexible and can be used with classes of an arbitrary level of granularity.

### 4.2 Datasets

The Oxford Advanced Learner’s Dictionary provides usage notes that contain typical predicates in various semantic domains in English, e.g., Traveling.<sup>9</sup> Each predicate is made up of a fixed part (e.g., a verb) and a generalizable part which contains one or more nouns.

Examples include *fix an election/the vote*, *bacteria/microbes/viruses spread*, *spend money/savings/a fortune*. In the case that more than one noun was provided, we split the textual expression into as many items as the number of nouns. For instance, from *spend money/savings/a fortune* we created three items in our dataset, i.e., *spend money*, *spend savings*, *spend a fortune*. The splitting procedure generated 6,220 instantiated lexical predicate items overall.

<sup>8</sup><http://wordnetcode.princeton.edu/standoff-files/core-wordnet.txt>

<sup>9</sup>[http://oald8.oxfordlearnersdictionaries.com/usage\\_notes/unbox\\_colloc/](http://oald8.oxfordlearnersdictionaries.com/usage_notes/unbox_colloc/)

k	Prec@k	Correct	Total
1	0.94	46	49
2	0.87	85	98
3	0.86	124	145
4	0.83	160	192
5	0.82	194	237
6	0.81	228	282
7	0.80	261	326
8	0.78	288	370
9	0.77	318	414
10	0.76	349	458
11	0.75	379	502
12	0.75	411	546
13	0.75	445	590
14	0.76	479	634
15	0.75	510	678
16	0.75	544	721
17	0.76	577	763
18	0.76	612	806
19	0.76	643	849
20	0.75	671	892

Table 3: Precision@ $k$  for ranking the semantic classes of lexical predicates.

### 4.3 Evaluating the Semantic Class Ranking

**Dataset.** Given the above dataset, we generalized each item by pairing its fixed verb part with \* (i.e., we keep “verb predicates” only, since they are more informative). For instance, the three items *bacteria/microbes/viruses spread* were generalized into the lexical predicate \* *spread*. The total number of different lexical predicates obtained was 1,446, totaling 1,429 distinct verbs (note that the dataset might contain the lexical predicate \* *spread* as well as *spread* \*).<sup>10</sup>

**Methodology.** For each lexical predicate we calculated the conditional probability of each semantic class using Formula 1, resulting in a ranking of semantic classes. To evaluate the top ranking classes, we calculated precision@ $k$ , with  $k$  ranging from 1 to 20, by counting all applicable classes as correct, e.g.,  $location_n^1$  is a valid semantic class for *travel to* \* while  $emotion_n^1$  is not.

**Results.** We show in Table 3 the precision@ $k$  calculated over a random sample of 50 lexical predicates.<sup>11</sup> As can be seen, while the classes quality is pretty high with low values of  $k$ , performance gradually degrades as we let  $k$  increase. This is mostly due to the highly polysemous nature of the predicates selected (e.g., *occupy* \*, *leave* \*, *help* \*, *attain* \*, *live* \*, etc.). We note that high performance, attaining above 80%, can be achieved

<sup>10</sup>The low number of items per predicate is due to the original Oxford resource.

<sup>11</sup>One lexical predicate did not have any semantic class ranking.

by focusing up to the first 7 classes output by our system, with a 94% precision@1.

#### 4.4 Evaluating Classification Performance

**Dataset.** Starting from the lexical predicate items obtained as described in Section 4.2, we selected those items belonging to a random sample of 20 usage notes among those provided by the Oxford dictionary, totaling 3,245 items. We then manually tagged each item’s argument (e.g., *virus* in *viruses spread*) with the most suitable semantic class (e.g., *virus<sub>n</sub><sup>1</sup>*), obtaining a gold standard dataset for the evaluation of our argument classification algorithm (cf. Section 3.4).

**Methodology.** In this second evaluation we measure the accuracy of our method at assigning the most suitable semantic class to the argument of a lexical predicate item in our gold standard. We use three customary measures to determine the quality of the acquired semantic classes, i.e., precision, recall and F1. Precision is the number of items which are assigned the correct class (as evaluated by a human) over the number of items which are assigned a class by the system. Recall is the number of items which are assigned the correct class over the number of items to be classified. F1 is the harmonic mean of precision and recall.

**Tuning.** The only parameter to be tuned is the factor  $\alpha$  that we use to mix the two probabilities in Formula 3 (cf. Section 3.4). For tuning  $\alpha$  we used a held-out set of 8 verbs, randomly sampled from the lexical predicates not used in the dataset. We created a tuning set using the annotated arguments in Wikipedia for these verbs: we trained the model on 80% of the annotated lexical predicate arguments (i.e., the class probability estimates in Formula 1) and then applied the probability mixture (i.e., Formula 3) for classifying the remaining 20% of arguments. Finally, we calculated the performance in terms of precision, recall and F1 with 11 different values of  $\alpha \in \{0, 0.1, \dots, 1.0\}$ , achieving optimal performance with  $\alpha = 0.2$ .

**Results.** Table 4 shows the results on the semantic class assignments. Our system shows very high precision, above 85%, while at the same time attaining an adequate 68% recall. We also compared against a random baseline that randomly selects one out of all the candidate semantic classes for each item, achieving only moderate results. A subsequent error analysis revealed the common types of error produced by our system: terms for which we could not provide (1) any WordNet concept

Method	Precision	Recall	F1
SPred	85.61	68.01	75.80
Random	40.96	40.96	40.96

Table 4: Performance on semantic class assignment.

(e.g., *political corruption*) or (2) any candidate semantic class (e.g., *immune system*).

#### 4.5 Disambiguation heuristics impact

As a follow-up analysis, for each dataset we considered the impact of each disambiguation heuristic described in Section 3.2 according to how many times it was triggered. Starting from the entire set of 1,446 lexical predicates from the Oxford dictionary (see Section 4.3), we counted the number of argument triples  $(a, s, l)$  already disambiguated in Wikipedia (i.e.,  $l \neq \epsilon$ ) and those disambiguated thanks to our disambiguation strategies. Table 5 shows the statistics. We note that, while the amount of originally linked arguments is very low (about 2.5% of total), our strategies are able to considerably increase the size of the initial set of linked instances. The most effective strategies appear to be the *One sense per page* and the *Trust the inventory*, which contribute 26.16% and 31.33% of the total links, respectively.

Even though most of the triples (i.e., 68 out of almost 74 million) remain unlinked, the ratio of distinct arguments which we linked to WordNet is considerably higher, calculated as 3,723,979 linked arguments over 12,431,564 distinct arguments, i.e., about 30%.

### 5 Experiment 2: Comparison with Kozareva & Hovy (2010)

Due to the novelty of the task carried out by SPred, the resulting output may be compared with only a limited number of existing approaches. The most similar approach is that of Kozareva and Hovy (2010, K&H) who assign supertypes to the arguments of arbitrary relations, a task which resembles our semantic predicate ranking. We therefore performed a comparison on the quality of the most highly-ranked supertypes (i.e., semantic classes) using their dataset of 24 relation patterns (i.e., lexical predicates).

**Dataset.** The dataset contained 14 lexical predicates (e.g., *work for \** or *\* fly to*), 10 of which were expanded in order to semantify their left- and right-side arguments (e.g., *\* work for* and *work for \**); for the remaining 4 predicates just a single

Total triples	Linked in Wikipedia	One sense per page	One sense per lexical predicate	Trust the inventory	Not linked
73,843,415	1,795,608	1,433,634	533,946	1,716,813	68,363,414

Table 5: Statistics on argument triple linking for all the lexical predicates in the Oxford dataset.

k	Prec@k	Correct	Total
1	0.88	21	24
2	0.90	43	48
3	0.88	63	72
4	0.89	85	96
5	0.91	109	120
6	0.91	131	144
7	0.92	154	168
8	0.91	175	192
9	0.92	198	216
10	0.92	221	240
11	0.92	242	264
12	0.92	264	288
13	0.91	284	312
14	0.90	304	336
15	0.91	327	360
16	0.91	348	384
17	0.90	367	408
18	0.89	386	432
19	0.89	407	456
20	0.89	429	480

Table 6: Precision@ $k$  for the semantic classes of the relations of Kozareva and Hovy (2010).

side was generalized (e.g., *\*dress*). While most of the relations apply to persons as a supertype, our method could find arguments for each of them.

**Methodology.** We carried out the same evaluation as in Section 4.3. We calculated precision@ $k$  of the semantic classes obtained for each relation in the dataset of K&H. Because the set of applicable classes was potentially unbounded, we were not able to report recall directly.

**Results.** K&H reported an overall accuracy of the top-20 superclasses of 92%. As can be seen in Table 6 we exhibit very good performance with increasing values of  $k$ . A comparison of Table 3 with Table 6 shows considerable differences in performance between the two datasets. We attribute this difference to the higher average WordNet polysemy of the verbal component of the Oxford predicates (on average 2.64 senses for K&H against 6.52 for the Oxford dataset).

Although we cannot report recall, we list the number of Wikipedia arguments and associated classes in Table 7, which provides an estimate of the extraction capability of SPred. The large number of classes found for the arguments demonstrates the ability of our method to generalize to a variety of semantic classes.

Predicate	Number of args	Number of classes
cause *	181,401	1,339
live in *	143,628	600
go to *	134,712	867
* cause	92,160	1,244
work in *	79,444	770
* go to	71,794	746
* live in	61,074	541
work on *	58,760	840
work for *	58,332	681
work at *	31,904	511
* work in	24,933	528
* celebrate	23,333	408

Table 7: Number of arguments and associated classes for the 12 most frequent lexical predicates of Kozareva and Hovy (2010) extracted by SPred from Wikipedia.

## 6 Related work

The availability of Web-scale corpora has led to the production of large resources of relations (Etzioni et al., 2005; Yates et al., 2007; Wu and Weld, 2010; Carlson et al., 2010; Fader et al., 2011). However, these resources often operate purely at the lexical level, providing no information on the semantics of their arguments or relations. Several studies have examined adding semantics through grouping relations into sets (Yates and Etzioni, 2009), ontologizing the arguments (Chklovski and Pantel, 2004), or ontologizing the relations themselves (Moro and Navigli, 2013). However, analysis has largely been either limited to ontologizing a small number of relation types with a fixed inventory, which potentially limits coverage, or has used implicit definitions of semantic categories (e.g., clusters of arguments), which limits interpretability. For example, Mohamed et al. (2011) use the semantic categories of the NELL system (Carlson et al., 2010) to learn roughly 400 valid ontologized relations from over 200M web pages, whereas WiSeNet (Moro and Navigli, 2012) leverages Wikipedia to acquire relation synsets for an open set of relations. Despite these efforts, no large-scale resource has existed to date that contains ontologized lexical predicates. In contrast, the present work provides a high-coverage method for learning argument superclasses from a broad-coverage ontology (WordNet), which can potentially be leveraged in relation extraction to ontolo-

gize relation arguments.

Our method for identifying the different semantic classes of predicate arguments is closely related to the task of identifying selectional preferences. The most similar approaches to it are taxonomy-based ones, which leverage the semantic types of the relations arguments (Resnik, 1996; Li and Abe, 1998; Clark and Weir, 2002; Pennacchiotti and Pantel, 2006). Nevertheless, despite their high quality sense-tagged data, these methods have often suffered from lack of coverage. As a result, alternative approaches have been proposed that eschew taxonomies in favor of rating the quality of potential relation arguments (Erk, 2007; Chambers and Jurafsky, 2010) or generating probability distributions over the arguments (Rooth et al., 1999; Pantel et al., 2007; Bergsma et al., 2008; Ritter et al., 2010; Séaghdha, 2010; Bouma, 2010; Jang and Mostow, 2012) in order to obtain higher coverage of preferences.

In contrast, we overcome the data sparsity of class-based models by leveraging the large quantity of collaboratively-annotated Wikipedia text in order to connect predicate arguments with their semantic class in WordNet using BabelNet (Navigli and Ponzetto, 2012); because we map directly to WordNet synsets, we provide a more readily-interpretable collocation preference model than most similarity-based or probabilistic models.

Verb frame extraction (Green et al., 2004) and predicate-argument structure analysis (Surdeanu et al., 2003; Yakushiji et al., 2006) are two areas that are also related to our work. But their generality goes beyond our intentions, as we focus on semantic predicates, which is much simpler and free from syntactic parsing.

Another closely related work is that of Hanks (2013) concerning the Theory of Norms and Exploitations, where norms (exploitations) represent expected (unexpected) classes for a given lexical predicate. Although our semantified predicates do, indeed, provide explicit evidence of norms obtained from collective intelligence and would provide support for this theory, exploitations present a more difficult task, different from the one addressed here, due to its focus on identifying property transfer between the semantic class and the exploited instance.

The closest technical approach to ours is that of Kozareva and Hovy (2010), who use recursive patterns to induce semantic classes for the arguments of relational patterns. Whereas their approach requires both a relation pattern and one or more seeds, which bias the types of semantic classes that are learned, our proposed method re-

quires only the pattern itself, and as a result is capable of learning an unbounded number of different semantic classes.

## 7 Conclusions

In this paper we present SPred, a novel approach to large-scale harvesting of semantic predicates. In order to semantify lexical predicates we exploit the wide coverage of Wikipedia to extract and disambiguate lexical predicate occurrences, and leverage WordNet to populate the semantic classes with suitable predicate arguments. As a result, we are able to ontologize lexical predicate instances like those available in existing dictionaries (e.g., *break a toe*) into semantic predicates (such as *break* a BODY PART).

For each lexical predicate (such as *break* \*), our method produces a probability distribution over the set of semantic classes (thus covering the different expected meanings for the filling arguments) and is able to classify new instances with the most suitable class. Our experiments show generally high performance, also in comparison with previous work on argument supertyping.

We hope that our semantic predicates will enable progress in different Natural Language Processing tasks such as Word Sense Disambiguation (Navigli, 2009), Semantic Role Labeling (Fürstenau and Lapata, 2012) or even Textual Entailment (Stern and Dagan, 2012) – each of which is in urgent need of reliable semantics. While we focused on semantifying lexical predicates, as future work we will apply our method to the ontologization of large amounts of sequences of words, such as phrases or textual relations (e.g., considering Google n-grams appearing in Wikipedia). Notably, our method should, in principle, generalize to any semantically-annotated corpus (e.g., Wikipedias in other languages), provided lexical predicates can be extracted with associated semantic classes.

In order to support future efforts we are releasing our semantic predicates as a freely available resource.<sup>12</sup>

## Acknowledgments

The authors gratefully acknowledge the support of the ERC Starting Grant MultiJEDI No. 259234.  

Thanks go to David A. Jurgens, Silvia Neçşulescu, Stefano Faralli and Moreno De Vincenzi for their help.

<sup>12</sup><http://lcl.uniroma1.it/spred>

## References

- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2012. Learning entailment relations by global graph structure optimization. *Computational Linguistics*, 38(1):73–111.
- Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *Proc. of EMNLP*, pages 59–68, Stroudsburg, PA, USA.
- Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia - a crystallization point for the Web of Data. *Web Semantics*, 7(3):154–165.
- Gerlof Bouma. 2010. Collocation Extraction beyond the Independence Assumption. In *Proc. of ACL, Short Papers*, pages 109–114, Uppsala, Sweden.
- Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proc. of AAAI*, pages 1306–1313, Atlanta, Georgia.
- Nathanael Chambers and Dan Jurafsky. 2010. Improving the use of pseudo-words for evaluating selectional preferences. In *Proc. of ACL*, pages 445–453, Stroudsburg, PA, USA.
- Tim Chklovski and Patrick Pantel. 2004. VerbOcean: Mining the Web for fine-grained semantic verb relations. In *Proc. of EMNLP*, pages 33–40, Barcelona, Spain.
- Jennifer Chu-Carroll and John Prager. 2007. An experimental study of the impact of information extraction accuracy on semantic search performance. In *Proc. of CIKM*, pages 505–514, Lisbon, Portugal.
- Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-Coverage Sense Disambiguation and Information Extraction with a Supersense Sequence Tagger. In *Proc. of EMNLP*, pages 594–602, Sydney, Australia.
- Stephen Clark and David Weir. 2002. Class-based probability estimation using a semantic hierarchy. *Computational Linguistics*, 28(2):187–206.
- Jonathan Crowther, editor. 1998. *Oxford Advanced Learner's Dictionary*. Cornelsen & Oxford, 5th edition.
- Flavio De Benedictis, Stefano Faralli, and Roberto Navigli. 2013. GlossBoot: Bootstrapping multilingual domain glossaries from the Web. In *Proc. of ACL*, Sofia, Bulgaria.
- Gerard de Melo and Gerhard Weikum. 2010. MENTA: Inducing Multilingual Taxonomies from Wikipedia. In *Proc. of CIKM*, pages 1099–1108, New York, NY, USA.
- Katrin Erk and Diana McCarthy. 2009. Graded word sense assignment. In *Proc. of EMNLP*, pages 440–449, Stroudsburg, PA, USA.
- Katrin Erk. 2007. A Simple, Similarity-based Model for Selectional Preferences. In *Proc. of ACL*, pages 216–223, Prague, Czech Republic.
- Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Un-supervised named-entity extraction from the web: an experimental study. *Artificial Intelligence*, 165(1):91–134.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *Proc. of EMNLP*, pages 1535–1545, Edinburgh, UK.
- Stefano Faralli and Roberto Navigli. 2013. A Java framework for multilingual definition and hypernym extraction. In *Proc. of ACL, Comp. Volume*, Sofia, Bulgaria.
- Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Database*. MIT Press, Cambridge, MA.
- David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building Watson: an overview of the DeepQA project. *AI Magazine*, 31(3):59–79.
- Hagen Fürstenau and Mirella Lapata. 2012. Semi-supervised semantic role labeling via structural alignment. *Computational Linguistics*, 38(1):135–171.
- Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proc. of HLT-NAACL*, pages 1–8, Edmonton, Canada.
- Rebecca Green, Bonnie J. Dorr, and Philip Resnik. 2004. Inducing Frame Semantic Verb Classes from WordNet and LDOCE. In *Proc. of ACL*, pages 375–382, Barcelona, Spain.
- Patrick Hanks. 2013. *Lexical Analysis: Norms and Exploitations*. University Press Group Limited.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of COLING*, pages 539–545, Nantes, France.
- Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61.
- Eduard H. Hovy, Roberto Navigli, and Simone Paolo Ponzetto. 2013. Collaboratively built semi-structured content and artificial intelligence: The story so far. *Artificial Intelligence*, 194:2–27.
- Ruihong Huang and Ellen Riloff. 2010. Inducing Domain-Specific Semantic Class Taggers from (Almost) Nothing. In *Proc. of ACL*, pages 275–285, Uppsala, Sweden.
- Sean P. Igo and Ellen Riloff. 2009. Corpus-based semantic lexicon induction with Web-based corroboration. In *Proc. of UMSLLS*, pages 18–26, Stroudsburg, PA, USA.
- Rubén Izquierdo, Armando Suárez, and German Rigau. 2009. An Empirical Study on Class-Based Word Sense Disambiguation. In *Proc. of EACL*, pages 389–397, Athens, Greece.
- Hyeju Jang and Jack Mostow. 2012. Inferring selectional preferences from part-of-speech n-grams. In *Proc. of EACL*, pages 377–386, Stroudsburg, PA, USA.

- Boris Katz, Jimmy J. Lin, Daniel Loreto, Wesley Hildebrandt, Matthew W. Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. 2003. Integrating Web-based and Corpus-based Techniques for Question Answering. In *Proc. of TREC*, pages 426–435, Gaithersburg, Maryland.
- Zornitsa Kozareva and Eduard Hovy. 2010. Learning Arguments and Supertypes of Semantic Relations Using Recursive Patterns. In *Proc. of ACL*, pages 1482–1491, Uppsala, Sweden.
- Zornitsa Kozareva, Ellen Riloff, and Eduard H. Hovy. 2008. Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. In *Proc. ACL/HLT*, pages 1048–1056, Columbus, Ohio.
- Sebastian Krause, Hong Li, Hans Uszkoreit, and Feiyu Xu. 2012. Large-scale learning of relation-extraction rules with distant supervision from the web. In *Proc. of ISWC 2012, Part I*, pages 263–278, Boston, MA.
- Hang Li and Naoki Abe. 1998. Generalizing case frames using a thesaurus and the MDL principle. *Computational Linguistics*, 24(2):217–244.
- Rada Mihalcea and Dan Moldovan. eXtended WordNet: Progress report. In *Proceedings of the NAACL-01 Workshop on WordNet and Other Lexical Resources*, Pittsburgh, Penn.
- Thahir Mohamed, Estevam Hruschka, and Tom Mitchell. 2011. Discovering Relations between Noun Categories. In *Proc. of EMNLP*, pages 1447–1455, Edinburgh, Scotland, UK.
- Andrea Moro and Roberto Navigli. 2012. WiSeNet: Building a Wikipedia-based semantic network with ontologized relations. In *Proc. of CIKM*, pages 1672–1676, Maui, HI, USA.
- Andrea Moro and Roberto Navigli. 2013. Integrating Syntactic and Semantic Analysis into the Open Information Extraction Paradigm. In *Proc. of IJCAI*, Beijing, China.
- Vivi Nastase and Michael Strube. 2013. Transforming wikipedia into a large scale multilingual concept network. *Artificial Intelligence*, 194:62–85.
- Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- Roberto Navigli and Paola Velardi. 2010. Learning Word-Class Lattices for Definition and Hypernym Extraction. In *Proc. of ACL*, pages 1318–1327, Uppsala, Sweden.
- Roberto Navigli. 2009. Word Sense Disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.
- Patrick Pantel, Rahul Bhagat, Timothy Chklovski, and Eduard Hovy. 2007. ISP: learning inferential selectional preferences. In *Proc. of NAACL*, pages 564–571, Rochester, NY.
- Marius Pasca. 2004. Acquisition of categorized named entities for web search. In *Proc. of CIKM*, pages 137–145, New York, NY, USA.
- Marco Pennacchiotti and Patrick Pantel. 2006. Ontologizing semantic relations. In *Proc. of COLING*, pages 793–800, Sydney, Australia.
- Simone Paolo Ponzetto and Michael Strube. 2011. Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence*, 175(9-10):1737–1756.
- Philip Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, 61(1):127–159.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A latent dirichlet allocation method for selectional preferences. In *Proc. of ACL*, pages 424–434, Uppsala, Sweden. ACL.
- Mats Rooth, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a semantically annotated lexicon via EM-based clustering. In *Proc. of ACL*, pages 104–111, Stroudsburg, PA, USA.
- Diarmuid O Séaghdha. 2010. Latent variable models of selectional preference. In *Proc. of ACL*, pages 435–444, Uppsala, Sweden.
- Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning Syntactic Patterns for Automatic Hypernym Discovery. In *NIPS*, pages 1297–1304, Cambridge, Mass.
- Asher Stern and Ido Dagan. 2012. Biutee: A modular open-source system for recognizing textual entailment. In *Proc. of ACL 2012, System Demonstrations*, pages 73–78, Jeju Island, Korea.
- Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proc. ACL*, pages 8–15, Stroudsburg, PA, USA.
- M. Thelen and E. Riloff. 2002. A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts. In *Proc. of EMNLP*, pages 214–221, Salt Lake City, UT, USA.
- Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. OntoLearn Reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3).
- Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial Intelligence*, 6(1):53–74.
- Fei Wu and Daniel S. Weld. 2010. Open Information Extraction Using Wikipedia. In *Proc. of ACL*, pages 118–127, Uppsala, Sweden.
- Akane Yakushiji, Yusuke Miyao, Tomoko Ohta, Yuka Tateisi, and Jun’ichi Tsujii. 2006. Automatic construction of predicate-argument structure patterns for biomedical information extraction. In *Proc. of EMNLP*, pages 284–292, Stroudsburg, PA, USA.
- David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. of ACL*, pages 189–196, Cambridge, MA, USA.
- Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research*, 34(1):255.
- Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. TextRunner: open information extraction on the web. In *Proc. of NAACL-Demonstrations*, pages 25–26, Stroudsburg, PA, USA.

# Towards Robust Abstractive Multi-Document Summarization: A Caseframe Analysis of Centrality and Domain

**Jackie Chi Kit Cheung**

University of Toronto  
10 King's College Rd., Room 3302  
Toronto, ON, Canada M5S 3G4  
jcheung@cs.toronto.edu

**Gerald Penn**

University of Toronto  
10 King's College Rd., Room 3302  
Toronto, ON, Canada M5S 3G4  
gpenn@cs.toronto.edu

## Abstract

In automatic summarization, *centrality* is the notion that a summary should contain the core parts of the source text. Current systems use centrality, along with redundancy avoidance and some sentence compression, to produce mostly extractive summaries. In this paper, we investigate how summarization can advance past this paradigm towards robust abstraction by making greater use of the domain of the source text. We conduct a series of studies comparing human-written model summaries to system summaries at the semantic level of *caseframes*. We show that model summaries (1) are more abstractive and make use of more sentence aggregation, (2) do not contain as many topical caseframes as system summaries, and (3) cannot be reconstructed solely from the source text, but can be if texts from in-domain documents are added. These results suggest that substantial improvements are unlikely to result from better optimizing centrality-based criteria, but rather more domain knowledge is needed.

## 1 Introduction

In automatic summarization, *centrality* has been one of the guiding principles for content selection in extractive systems. We define centrality to be the idea that a summary should contain the parts of the source text that are most similar or representative of the source text. This is most transparently illustrated by the Maximal Marginal Relevance (MMR) system of Carbonell and Goldstein (1998), which defines the summarization objective

to be a linear combination of a centrality term and a non-redundancy term.

Since MMR, much progress has been made on more sophisticated methods of measuring centrality and integrating it with non-redundancy (See Nenkova and McKeown (2011) for a recent survey). For example, term weighting methods such as the signature term method of Lin and Hovy (2000) pick out salient terms that occur more often than would be expected in the source text based on frequencies in a background corpus. This method is a core component of the most successful summarization methods (Conroy et al., 2006).

While extractive methods based on centrality have thus achieved success, there has long been recognition that abstractive methods are ultimately more desirable. One line of work is in text simplification and sentence fusion, which focus on the ability of abstraction to achieve a higher compression ratio (Knight and Marcu, 2000; Barzilay and McKeown, 2005). A less examined issue is that of aggregation and information synthesis. A key part of the usefulness of summaries is that they provide some synthesis or analysis of the source text and make a more general statement that is of direct relevance to the user. For example, a series of related events can be aggregated and expressed as a trend.

The position of this paper is that centrality is not enough to make substantial progress towards abstractive summarization that is capable of this type of semantic inference. Instead, summarization systems need to make more use of domain knowledge. We provide evidence for this in a series of studies on the TAC 2010 guided summarization data set that examines how the behaviour of automatic summarizers can or cannot be distinguished from human summarizers. First, we confirm that abstraction is a desirable goal, and

provide a quantitative measure of the degree of sentence aggregation in a summarization system. Second, we show that centrality-based measures are unlikely to lead to substantial progress towards abstractive summarization, because current top-performing systems already produce summaries that are more “central” than humans do. Third, we consider how domain knowledge may be useful as a resource for an abstractive system, by showing that key parts of model summaries can be reconstructed from the source plus related in-domain documents.

Our contributions are novel in the following respects. First, our analyses are performed at the level of *caseframes*, rather at the level of words or syntactic dependencies as in previous work. Caseframes are shallow approximations of semantic roles which are well suited to characterizing a domain by its slots. Furthermore, we take a *developmental* rather than *evaluative* perspective—our goal is not to develop a new evaluation measure as defined by correlation with human responsiveness judgments. Instead, our studies reveal useful criteria with which to distinguish human-written and system summaries, helping to guide the development of future summarization systems.

## 2 Related Work

Domain-dependent template-based summarization systems have been an alternative to extractive systems which make use of rich knowledge about a domain and information extraction techniques to generate a summary, possibly using a natural language generation system (Radev and McKeown, 1998; White et al., 2001; McKeown et al., 2002). This paper can be seen as a first step towards reconciling the advantages of domain knowledge with the resource-lean extraction approaches popular today.

As noted above, Lin and Hovy’s (2000) signature terms have been successful in discovering terms that are specific to the source text. These terms are identified by a log-likelihood ratio test based on their relative frequencies in relevant and irrelevant documents. They were originally proposed in the context of single-document summarization, where they were calculated using in-domain (relevant) vs. out-of-domain (irrelevant) text. In multi-document summarization, the in-domain text has been replaced by the source text cluster (Conroy et al., 2006), thus they are now

used as a form of centrality-based features. In this paper, we use guided summarization data as an opportunity to reopen the investigation into the effect of domain, because multiple document clusters from the same domain are available.

Summarization evaluation is typically done by comparing system output to human-written model summaries, and are validated by their correlation with user responsiveness judgments. The comparison can be done at the word level, as in ROUGE (Lin, 2004), at the syntactic level, as in Basic Elements (Hovy et al., 2006), or at the level of summary content units, as in the Pyramid method (Nenkova and Passonneau, 2004). There are also automatic measures which do not require model summaries, but compare against the source text instead (Louis and Nenkova, 2009; Saggion et al., 2010).

Several studies complement this paper by examining the best possible extractive system using current evaluation measures, such as ROUGE (Lin and Hovy, 2003; Conroy et al., 2006). They find that the best possible extractive systems score higher or as highly than human summarizers, but it is unclear whether this means the oracle summaries are actually as useful as human ones in an extrinsic setting. Genest et al. (2009) ask humans to create extractive summaries, and find that they score in between current automatic systems and human-written abstracts on responsiveness, linguistic quality, and Pyramid score. In the lecture domain, He et al. (1999; 2000) find that lecture transcripts that have been manually highlighted with key points improve students’ quiz scores more than when using automated summarization techniques or when providing only the lecture transcript or slides.

Jing and McKeown (2000) manually analyzed 30 human-written summaries, and find that 19% of sentences cannot be explained by *cut-and-paste* operations from the source text. Saggion and Lapalme (2002) similarly define a list of transformations necessary to convert source text to summary text, and manually analyzed their frequencies. Copeck and Szpakowicz (2004) find that at most 55% of vocabulary items found in model summaries occur in the source text, but they do not investigate where the other vocabulary items might be found.

<b>Sentence:</b>	
<i>At one point, two bomb squad trucks sped to the school after a backpack scare.</i>	
<b>Dependencies:</b>	
<i>num(point, one)</i>	<i>prep_at(sped, point)</i>
<i>num(trucks, two)</i>	<i>nn(trucks, bomb)</i>
<i>nn(trucks, squad)</i>	<i>nsubj(sped, trucks)</i>
<i>root(ROOT, sped)</i>	<i>det(school, the)</i>
<i>prep_to(sped, school)</i>	<i>det(scare, a)</i>
<i>nn(scare, backpack)</i>	<i>prep_after(sped, scare)</i>
<b>Caseframes:</b>	
<i>(speed, prep_at)</i>	<i>(speed, nsubj)</i>
<i>(speed, prep_to)</i>	<i>(speed, prep_after)</i>

Table 1: A sentence decomposed into its dependency edges, and the caseframes derived from those edges that we consider (in black).

### 3 Theoretical basis of our analysis

Many existing summarization evaluation methods rely on word or N-gram overlap measures, but these measures are not appropriate for our analysis. Word overlap can occur due to shared proper nouns or entity mentions. Good summaries should certainly contain the salient entities in the source text, but when assessing the effect of the domain, different domain instances (i.e., different document clusters in the same domain) would be expected to contain different salient entities. Also, the realization of entities as noun phrases depends strongly on context, which would confound our analysis if we do not also correctly resolve coreference, a difficult problem in its own right. We leave such issues to other work (Nenkova and McKeown, 2003, e.g.).

Domains would rather be expected to share *slots* (a.k.a. *aspects*), which require a more semantic level of analysis that can account for the various ways in which a particular slot can be expressed. Another consideration is that the structures to be analyzed should be extracted automatically. Based on these criteria, we selected *caseframes* to be the appropriate unit of analysis. A caseframe is a shallow approximation of the semantic role structure of a proposition-bearing unit like a verb, and are derived from the dependency parse of a sentence<sup>1</sup>.

<sup>1</sup>Note that caseframes are distinct from (though directly

<b>Relation</b>	<b>Caseframe Pair</b>	<b>Sim.</b>
Degree	<i>(kill, dobj)</i> <i>(wound, dobj)</i>	0.82
Causal	<i>(kill, dobj)</i> <i>(die, nsubj)</i>	0.80
Type	<i>(rise, dobj)</i> <i>(drop, prep_to)</i>	0.81

Figure 1: Sample pairs of similar caseframes by relation type, and the similarity score assigned to them by our distributional model.

In particular, they are *(gov, role)* pairs, where *gov* is a proposition-bearing element, and *role* is an approximation of a semantic role with *gov* as its head (See Figure 1 for examples). Caseframes do not consider the dependents of the semantic role approximations.

The use of caseframes is well grounded in a variety of NLP tasks relevant to summarization such as coreference resolution (Bean and Riloff, 2004), and information extraction (Chambers and Jurafsky, 2011), where they serve the central unit of semantic analysis. Related semantic representations are popular in Case Grammar and its derivative formalisms such as frame semantics (Fillmore, 1982).

We use the following algorithm to extract caseframes from dependency parses. First, we extract those dependency edges with a relation type of subject, direct object, indirect object, or prepositional object (with the preposition indicated), along with their governors. The governor must be a verb, event noun (as defined by the hyponyms of the WordNet EVENT synset), or nominal or adjectival predicate. Then, a series of deterministic transformations are applied to the syntactic relations to account for voicing alternations, control, raising, and copular constructions.

#### 3.1 Caseframe Similarity

Direct caseframe matches account for some variation in the expression of slots, such as voicing alternations, but there are other reasons different caseframes may indicate the same slot (Figure 1). For example, *(kill, dobj)* and *(wound, dobj)* both indicate the victim of an attack, but differ by the degree of injury to the victim. *(kill, dobj)* and *(die, nsubj)* also refer to a victim, but are linked by a causal relation. *(rise, dobj)* and

inspired by) the similarly named *case frames* of Case Grammar (Fillmore, 1968).

(*drop, prep\_to*) on the other hand simply share a named entity type (in this case, numbers). To account for these issues, we measure caseframe similarity based on their distributional similarity in a large training corpus.

First, we construct vector representations of each caseframe, where the dimensions of the vector correspond to the lemma of the head word that fills the caseframe in the training corpus. For example, *kicked the ball* would result in a count of 1 added to the caseframe (*kick, dobj*) for the context word *ball*. Then, we rescale the counts into pointwise mutual information values, which has been shown to be more effective than raw counts at detecting semantic relatedness (Turney, 2001). Similarity between caseframes can then be compared by cosine similarity between their vector representations.

For training, we use the AFP portion of the Gigaword corpus (Graff et al., 2005), which we parsed using the Stanford parser’s typed dependency tree representation with collapsed conjunctions (de Marneffe et al., 2006). For reasons of sparsity, we only considered caseframes that appear at least five times in the guided summarization corpus, and only the 3000 most common lemmata in Gigaword as context words.

### 3.2 An Example

To illustrate how caseframes indicate the slots in a summary, we provide the following fragment of a model summary from TAC about the *Unabomber trial*:

- (1) *In Sacramento, Theodore Kaczynski faces a 10-count federal indictment for 4 of the 16 mail bomb attacks attributed to the Unabomber in which two people were killed. If found guilty, he faces a death penalty. ... He has pleaded innocent to all charges. U.S. District Judge Garland Burrell Jr. presides in Sacramento.*

All of the slots provided by TAC for the *Investigations and Trials* domain can be identified by one or more caseframes. The DEFENDANT can be identified by (*face, nsubj*), and (*plead, nsubj*); the CHARGES by (*face, dobj*); the REASON by (*indictment, prep\_for*); the SENTENCE by (*face, dobj*); the PLEAD by (*plead, dobj*); and the INVESTIGATOR by (*preside, nsubj*).

## 4 Experiments

We conducted our experiments on the data and results of the TAC 2010 summarization workshop. This data set contains 920 newspaper articles in 46 topics of 20 documents each. Ten are used in an initial guided summarization task, and ten are used in an update summarization task, in which a summary must be produced assuming that the original ten documents had already been read. All summaries have a word length limit of 100 words. We analyzed the results of the two summarization tasks separately in our experiments.

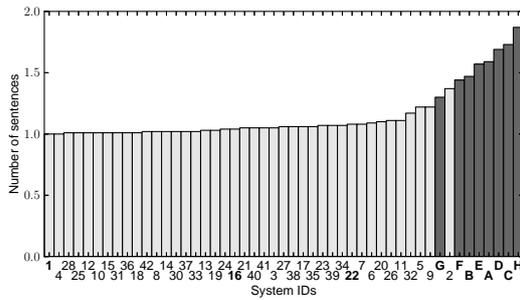
The 46 topics belong to five different categories or domains: *Accidents and natural disasters*, *Criminal or terrorist attacks*, *Health and safety*, *Endangered resources*, and *Investigations and trials*. Each domain is associated with a template specifying the type of information that is expected in the domain, such as the participants in the event or the time that the event occurred.

In our study, we compared the characteristics of summaries generated by the eight human summarizers with those generated by the peer summaries, which are basically extractive systems. There are 43 peer summarization systems, including two baselines defined by NIST. We refer to systems by their ID given by NIST, which are alphabetical for the human summarizers (A to H), and numeric for the peer summarizers (1 to 43). We removed two peer systems (systems 29 and 43) which did not generate any summary text in the workshop, presumably due to software problems. For each measure that we consider, we compare the average among the human-written summaries to the three individual peer systems, which we chose in order to provide a representative sample of the average and best performance of the automatic systems according to current evaluation methods. These systems are all primarily extractive, like most of the systems in the workshop:

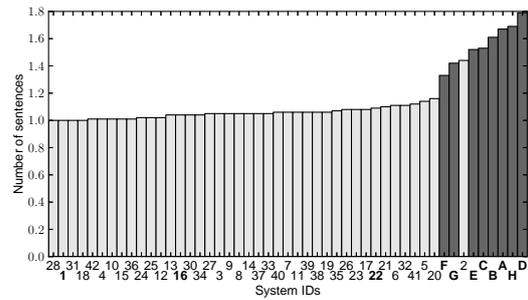
**Peer average** The average of the measure among the 41 peer summarizers.

**Peer 16** This system scored the highest in responsiveness scores on the original summarization task and in ROUGE-2, responsiveness, and Pyramid score in the update task.

**Peer 22** This system scored the highest in ROUGE-2 and Pyramid score in the original summarization task.



(a) Initial guided summarization task



(b) Update summarization task

Figure 2: Average sentence cover size: the average number of sentences needed to generate the caseframes in a summary sentence (Study 1). Model summaries are shown in darker bars. Peer system numbers that we focus on are in bold.

Condition	Initial	Update
Model average	1.58	1.57
Peer average	1.06	1.06
Peer 1	1.00	1.00
Peer 16	1.04	1.04
Peer 22	1.08	1.09

Table 2: The average number of source text sentences needed to cover a summary sentence. The model average is statistically significantly different from all the other conditions  $p < 10^{-7}$  (Study 1).

**Peer 1** The NIST-defined baseline, which is the leading sentence baseline from the most recent document in the source text cluster. This system scored the highest on linguistic quality in both tasks.

#### 4.1 Study 1: Sentence aggregation

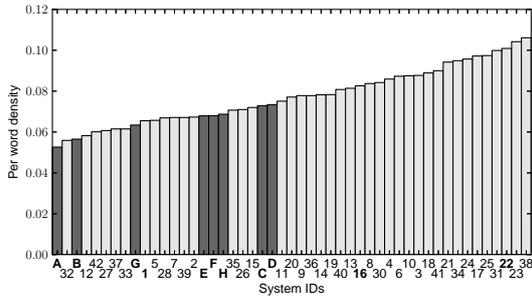
We first confirm that human summarizers are more prone to sentence aggregation than system summarizers, showing that abstraction is indeed a desirable goal. To do so, we propose a measure to quantify the degree of sentence aggregation exhibited by a summarizer, which we call **average sentence cover size**. This is defined to be the minimum number of sentences from the source text needed to cover all of the caseframes found in a summary sentence (for those caseframes that can be found in the source text at all), averaged over all of the summary sentences. Purely extractive systems would thus be expected to score 1.0, as would systems that perform text compression by remov-

ing constituents of a source text sentence. Human summarizers would be expected to score higher, if they actually aggregate information from multiple points in the source text.

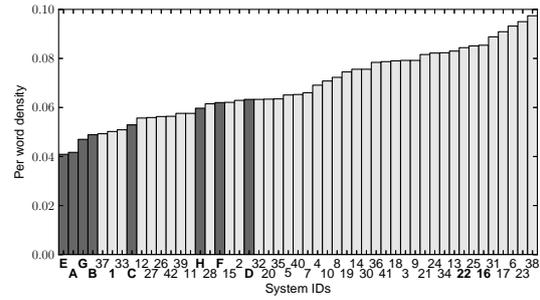
To illustrate, suppose we assign arbitrary indices to caseframes, a summary sentence contains caseframes  $\{1, 2, 3, 4, 5\}$ , and the source text contains three sentences with caseframes, which can be represented as a nested set  $\{\{1, 3, 4\}, \{2, 5, 6\}, \{1, 4, 7\}\}$ . Then, the summary sentence can be covered by two sentences from the source text, namely  $\{\{1, 3, 4\}, \{2, 5, 6\}\}$ .

This problem is actually an instance of the minimum set cover problem, in which sentences are sets, and caseframes are set elements. Minimum set cover is NP-hard in general, but the standard integer programming formulation of set cover sufficed for our data set; we used ILOG CPLEX 12.4’s mixed integer programming mode to solve all the set cover problems optimally.

**Results** Figure 2 shows the ranking of the summarizers by this measure. Most peer systems have a low average sentence cover size of close to 1, which reflects the fact that they are purely or almost purely extractive. Human model summarizers show a higher degree of aggregation in their summaries. The averages of the tested conditions are shown in Table 2, and are statistically significant. Peer 2 shows a relatively high level of aggregation despite being an extractive system. Upon inspection of its summaries, it appears that Peer 2 tends to select many datelines, and without punctuation to separate them from the rest of the summary, our automatic analysis tools incorrectly merged many sentences together, resulting in incorrect parses and novel caseframes not found in



(a) Initial guided summarization task



(b) Update summarization task

Figure 3: Density of signature caseframes (Study 2).

<b>Topic: Unabomber trial</b>
<i>(charge, dobj), (kill, dobj),</i>
<i>(trial, prep_of), (bombing, prep_in)</i>
<b>Topic: Mangrove forests</b>
<i>(beach, prep_of), (save, dobj)</i>
<i>(development, prep_of), (recover, nsubj)</i>
<b>Topic: Bird Flu</b>
<i>(infect, prep_with), (die, nsubj)</i>
<i>(contact, dobj), (import, prep_from)</i>

Figure 4: Examples of signature caseframes found in Study 2.

the source text.

#### 4.2 Study 2: Signature caseframe density

Study 1 shows that human summarizers are more abstractive in that they aggregate information from multiple sentences in the source text, but how is this aggregation performed? One possibility is that human summary writers are able to pack a greater number of salient caseframes into their summaries. That is, humans are fundamentally relying on centrality just as automatic summarizers do, and are simply able to achieve higher compression ratios by being more succinct. If this is true, then sentence fusion methods over the source text alone might be able to solve the problem. Unfortunately, we show that this is false and that system summaries are actually more central than model ones.

To extract topical caseframes, we use Lin and Hovy’s (2000) method of calculating signature terms, but extend the method to apply it at the caseframe rather than the word level. We follow Lin and Hovy (2000) in using a significance

Condition	Initial	Update
Model average	0.065	0.052
Peer average	0.080*	0.072*
Peer 1	0.066	0.050
Peer 16	0.083*	0.085*
Peer 22	0.101*	0.084*

Table 3: Signature caseframe densities for different sets of summarizers, for the initial and update guided summarization tasks (Study 2). \*:  $p < 0.005$ .

threshold of 0.001 to determine signature caseframes<sup>2</sup>. Figure 4 shows examples of signature caseframes for several topics. Then, we calculate the **signature caseframe density** of each of the summarization systems. This is defined to be the number of signature caseframes in the set of summaries divided by the number of words in that set of summaries.

**Results** Figure 3 shows the density for all of the summarizers, in ascending order of density. As can be seen, the human abstractors actually tend to use fewer signature caseframes in their summaries than automatic systems. Only the leading baseline is indistinguishable from the model average. Table 3 shows the densities for the conditions that we described earlier. The differences in density between the human average and the non-baseline conditions are highly statistically significant, according to paired two-tailed Wilcoxon signed-rank tests for the statistic calculated for each topic cluster.

These results show that human abstractors do

<sup>2</sup>We tried various other thresholds, but the results were much the same.

Threshold	0.9		0.8	
Condition	Init.	Up.	Init.	Up.
Model average	0.066	0.052	0.062	0.047
Peer average	0.080	0.071	0.071	0.063
Peer 1	0.068	0.050	0.060	0.044
Peer 16	0.083	0.086	0.072	0.077
Peer 22	0.100	0.086	0.084	0.075

Table 4: Density of signature caseframes after merging to various threshold for the initial (**Init.**) and update (**Up.**) summarization tasks (Study 2).

not merely repeat the caseframes that are indicative of a topic cluster or use minor grammatical alternations in their summaries. Rather, a genuine sort of abstraction or distillation has taken place, either through paraphrasing or semantic inference, to transform the source text into the final informative summary.

**Merging Caseframes** We next investigate whether simple paraphrasing could account for the above results; it may be the case that human summarizers simply replace words in the source text with synonyms, which can be detected with distributional similarity. Thus, we merged similar caseframes into clusters according to the distributional semantic similarity defined in Section 3.1, and then repeated the previous experiment. We chose two relatively high levels of similarity (0.8 and 0.9), and used complete-link agglomerative (i.e., bottom-up) clustering to merge similar caseframes. That is, each caseframe begins as a separate cluster, and the two most similar clusters are merged at each step until the desired similarity threshold is reached. Cluster similarity is defined to be the minimum similarity (or equivalently, maximum distance) between elements in the two clusters; that is,  $\max_{c \in C_1, c' \in C_2} \text{sim}(c, c')$ . Complete-link agglomerative clustering tends to form coherent clusters where the similarity between any pair within a cluster is high (Manning et al., 2008).

**Cluster Results** Table 4 shows the results after the clustering step, with similarity thresholds of 0.9 and 0.8. Once again, model summaries contain a lower density of signature caseframes. The statistical significance results are unchanged. This indicates that simple paraphrasing alone cannot account for the difference in the signature caseframe

densities, and that some deeper abstraction or semantic inference has occurred.

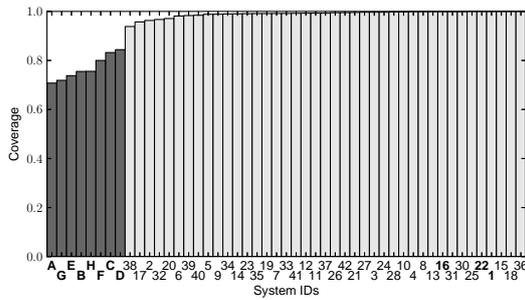
Note that we are not claiming that a lower density of signature caseframes necessarily correlates with a more informative summary. For example, some automatic summarizers are comparable to the human abstractors in their relatively low density of signature caseframes, but these turn out to be the lowest performing summarization systems by all measures in the workshop, and they are unlikely to rival human abstractors in any reasonable evaluation of summary informativeness. It does, however, appear that further optimizing centrality-based measures alone is unlikely to produce better informative summaries, even if we analyze the summary at a syntactic/semantic rather than lexical level.

### 4.3 Study 3: Summary Reconstruction

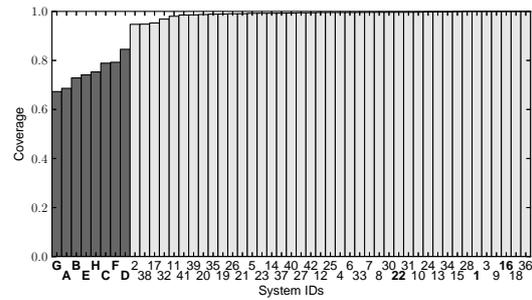
The above studies show that the higher degree of abstraction in model summaries cannot be explained by better compression of topically salient caseframes alone. We now switch perspectives to ask how model summaries might be automatically generated at all. We will show that they cannot be reconstructed solely from the source text, extending Copeck and Szpakowicz (2004)’s result to caseframes. However, we also show that if articles from the same domain are added, reconstruction then becomes possible. Our measure of whether a model summary can be reconstructed is **caseframe coverage**. We define this to be the proportion of caseframes in a summary that is contained by some reference set. This is thus a score between 0 and 1. Unlike in the previous study, we use the full set of caseframes, not just signature caseframes, because our goal now to create a hypothesis space from which it is in principle possible to generate the model summaries.

**Results** We first calculated caseframe coverage with respect to the source text alone (Figure 5). As expected, automatic systems show close to perfect coverage, because of their basically extractive nature, while model summaries show much lower coverage. These statistics are summarized by Table 5. These results present a fundamental limit to extractive systems, and also text simplification and sentence fusion methods based solely on the source text.

**The Impact of Domain Knowledge** How might automatic summarizers be able to acquire these



(a) Initial guided summarization task



(b) Update summarization task

Figure 5: Coverage of summary text caseframes in source text (Study 3).

Condition	Initial	Update
Model average	0.77	0.75
Peer average	0.99	0.99
Peer 1	1.00	1.00
Peer 16	1.00	1.00
Peer 22	1.00	1.00

Table 5: Coverage of caseframes in summaries with respect to the source text. The model average is statistically significantly different from all the other conditions  $p < 10^{-8}$  (Study 3).

caseframes from other sources? Traditional systems that perform semantic inference do so from a set of known facts about the domain in the form of a knowledge base, but as we have seen, most extractive summarization systems do not make much use of in-domain corpora. We examine adding in-domain text to the source text to see how this would affect coverage.

Recall that the 46 topics in TAC 2010 are categorized into five domains. To calculate the impact of domain knowledge, we add all the documents that belong in the same domain to the source text to calculate coverage. To ensure that coverage does not increase simply due to increasing the size of the reference set, we compare to the baseline of adding the same number of documents that belong to another domain. As shown in Table 6, the effect of adding more in-domain text on caseframe coverage is substantial, and noticeably more than using out-of-domain text. In fact, nearly all caseframes can be found in the expanded set of articles. The implication of this result is that it may be possible to generate better summaries by mining in-domain text for relevant caseframes.

Reference corpus	Initial	Update
Source text only	0.77	0.75
+out-of-domain	0.91	0.91
+in-domain	0.98	0.97

Table 6: The effect on caseframe coverage of adding in-domain and out-of-domain documents. The difference between adding in-domain and out-of-domain text is significant  $p < 10^{-3}$  (Study 3).

## 5 Conclusion

We have presented a series of studies to distinguish human-written informative summaries from the summaries produced by current systems. Our studies are performed at the level of caseframes, which are able to characterize a domain in terms of its slots. First, we confirm that model summaries are more abstractive and aggregate information from multiple source text sentences. Then, we show that this is not simply due to summary writers packing together source text sentences containing topical caseframes to achieve a higher compression ratio, even if paraphrasing is taken into account. Indeed, model summaries cannot be reconstructed from the source text alone. However, our results are also positive in that we find that nearly all model summary caseframes can be found in the source text together with some in-domain documents.

Current summarization systems have been heavily optimized towards centrality and lexical-semantic reasoning, but we are nearing the bottom of the barrel. Domain inference, on the other hand, and a greater use of in-domain documents as a knowledge source for domain inference, are very promising indeed. Mining useful caseframes

for a sentence fusion-based approach has the potential, as our experiments have shown, to deliver results in just the areas where current approaches are weakest.

## Acknowledgements

This work is supported by the Natural Sciences and Engineering Research Council of Canada.

## References

- Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.
- David Bean and Ellen Riloff. 2004. Unsupervised learning of contextual role knowledge for coreference resolution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*.
- Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336. ACM.
- Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 976–986, Portland, Oregon, USA, June. Association for Computational Linguistics.
- John M. Conroy, Judith D. Schlesinger, and Dianne P. O’Leary. 2006. Topic-focused multi-document summarization using an approximate oracle score. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 152–159, Sydney, Australia, July. Association for Computational Linguistics.
- Terry Copeck and Stan Szpakowicz. 2004. Vocabulary agreement among model summaries and source documents. In *Proceedings of the 2004 Document Understanding Conference (DUC)*.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC 2006*.
- Charles Fillmore. 1968. The case for case. In E. Bach and R. T. Harms, editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Reinhart, and Winston, New York.
- Charles J. Fillmore. 1982. Frame semantics. *Linguistics in the Morning Calm*, pages 111–137.
- Pierre-Etienne Genest, Guy Lapalme, and Mehdi Yousfi-Monod. 2009. Hextac: the creation of a manual extractive run. In *Proceedings of the Second Text Analysis Conference, Gaithersburg, Maryland, USA. National Institute of Standards and Technology*.
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2005. English gigaword second edition. *Linguistic Data Consortium, Philadelphia*.
- Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. 1999. Auto-summarization of audio-video presentations. In *Proceedings of the Seventh ACM International Conference on Multimedia*. ACM.
- Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. 2000. Comparing presentation summaries: slides vs. reading vs. listening. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI ’00*, pages 177–184, New York, NY, USA. ACM.
- Eduard Hovy, Chin-Yew Lin, Liang Zhou, and Junichi Fukumoto. 2006. Automated summarization evaluation with Basic Elements. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, pages 899–902.
- IBM. *IBM ILOG CPLEX Optimization Studio V12.4*.
- Hongyan Jing and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pages 178–185.
- Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. In *Proceedings of the National Conference on Artificial Intelligence*.
- Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 1, COLING ’00*, pages 495–501, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chin-Yew Lin and Eduard Hovy. 2003. The potential and limitations of automatic sentence extraction for summarization. In *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop*. Association for Computational Linguistics.
- Chin Y. Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In Stan Szpakowicz and Marie-Francine Moens, editors, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.
- Annie Louis and Ani Nenkova. 2009. Automatically evaluating content selection in summarization without human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language*

*Processing*. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, 2008. *Introduction to Information Retrieval*, chapter 17. Cambridge University Press.

Kathleen R. McKeown, Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith L. Klavans, Ani Nenkova, Carl Sable, Barry Schiffman, and Sergey Sigelman. 2002. Tracking and summarizing news on a daily basis with Columbia's Newsblaster. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 280–285. Morgan Kaufmann Publishers Inc.

Ani Nenkova and Kathleen McKeown. 2003. References to named entities: a corpus study. In *Companion Volume of the Proceedings of HLT-NAACL 2003 - Short Papers*. Association for Computational Linguistics.

Ani Nenkova and Kathleen McKeown. 2011. Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2):103–233.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, volume 2004, pages 145–152.

Dragomir R. Radev and Kathleen R. McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):470–500.

Horacio Saggion and Guy Lapalme. 2002. Generating indicative-informative summaries with SumUM. *Computational linguistics*, 28(4):497–526.

Horacio Saggion, Juan-Manuel Torres-Moreno, Iria Cunha, and Eric SanJuan. 2010. Multilingual summarization evaluation without human models. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1059–1067. Association for Computational Linguistics.

Peter Turney. 2001. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001)*, pages 491–502.

Michael White, Tanya Korelsky, Claire Cardie, Vincent Ng, David Pierce, and Kiri Wagstaff. 2001. Multidocument summarization via information extraction. In *Proceedings of the First International Conference on Human Language Technology Research*. Association for Computational Linguistics.

# HEADY: News headline abstraction through event pattern clustering

Enrique Alfonseca

Google Inc.

ealfonseca@google.com

Daniele Pighin

Google Inc.

biondo@google.com

Guillermo Garrido\*

NLP & IR Group at UNED

ggarrido@lsi.uned.es

## Abstract

This paper presents HEADY: a novel, *abstractive* approach for headline generation from news collections. From a web-scale corpus of English news, we mine syntactic patterns that a Noisy-OR model generalizes into event descriptions. At inference time, we query the model with the patterns observed in an unseen news collection, identify the event that better captures the gist of the collection and retrieve the most appropriate pattern to generate a headline. HEADY improves over a state-of-the-art open-domain title abstraction method, bridging half of the gap that separates it from extractive methods using human-generated titles in manual evaluations, and performs comparably to human-generated headlines as evaluated with ROUGE.

## 1 Introduction

**Motivation.** News events are rarely reported only in one way, from a single point of view. Different news agencies will interpret the event in different ways; various countries or locations may highlight different aspects of it depending on how they are affected; and opinions and in-depth analyses will be written after the fact.

The variety of contents and styles is both an opportunity and a challenge. On the positive side, we have the same events described in different ways; this redundancy is useful for summarization, as the information content reported by the majority of news sources most likely represents the central part of the event. On the other hand, variability and subjectivity can be difficult to isolate. For some applications it is important to understand, given a collection of related news articles and re-

- 
- Carmelo and La La Party It Up with Kim and Ciara
  - La La Vazquez and Carmelo Anthony: Wedding Day Bliss
  - Carmelo Anthony, actress LaLa Vazquez wed in NYC
  - Stylist to the Stars
  - LaLa, Carmelo Set Off Celebrity Wedding Weekend
  - Ciara rocks a sexy Versace Spring 2010 mini to LaLa Vasquez and Carmelo Anthony's wedding (photos)
  - Lala Vasquez on her wedding dress, cake, reality tv show and fiancé, Carmelo Anthony (video)
  - VAZQUEZ MARRIES SPORTS STAR ANTHONY
  - LeBron Returns To NYC For Carmelo's Wedding
  - Carmelo Anthony's stylist dishes on the wedding
  - Paul pitching another Big Three with "Melo in NYC"
  - Carmelo Anthony and La La Vazquez Get Married at Star-Studded Wedding Ceremony
- 

Table 1: Headlines observed for a news collection reporting the same wedding event.

ports, how to formulate in an objective way what has happened.

As a motivating example, Table 1 shows the different headlines observed in news reporting the wedding between basketball player Carmelo Anthony and actress LaLa Vazquez. As can be seen, there is a wide variety of ways to report the same event, including different points of view, highlighted aspects, and opinionated statements on the part of the reporter. When presenting this event to a user in a news-based information retrieval or recommendation system, different event descriptions may be more appropriate. For example, a user may only be interested in objective, informative summaries without any interpretation on the part of the reporter. In this case, *Carmelo Anthony, ac-*

\*Work done during an internship at Google Zurich.

*tress LaLa Vazquez wed in NYC* would be a good choice.

**Goal.** Our final goal in this research is to build a headline generation system that, given a news collection, is able to describe it with the most compact, objective and informative headline. In particular, we want the system to be able to:

- Generate headlines in an open-domain, unsupervised way, so that it does not need to rely on training data which is expensive to produce.
- Generalize across synonymous expressions that refer to the same event.
- Do so in an abstractive fashion, to enforce novelty, objectivity and generality.

In order to advance towards this goal, this paper explores the following questions:

- What is a good way of using syntactic patterns to represent events for generating headlines?
- Can we have satisfactory readability with an open-domain abstractive approach, not relying on training data nor on manually predefined generation templates?
- How far can we get in terms of informativeness, compared to the human-produced headlines, i.e., extractive approaches?

**Contributions.** In this paper we present HEADY, which is at the same time a novel system for abstractive headline generation, and a smooth clustering of patterns describing the same events. HEADY is fully open-domain and can scale to web-sized data. By learning to generalize events across the boundaries of a single news story or news collection, HEADY produces compact and effective headlines that objectively convey the relevant information.

When compared to a state-of-the-art open-domain headline abstraction system (Filippova, 2010), the new headlines are statistically significantly better both in terms of readability and informativeness. Also, automatic evaluations using ROUGE, having objective headlines for the news as references, show that the abstractive headlines are on par with human-produced headlines.

## 2 Related work

### Headline generation and summarization.

Most headline generation work in the past has focused on the problem of single-document summarization: given the main passage of a single news article, generate a very short summary of the article. From early in the field, it was pointed out that a purely extractive approach is not good enough to generate headlines from the body text (Banko et al., 2000). Sometimes the most important information is distributed across several sentences in the document. More importantly, quite often, the single sentence selected as the most informative for the news collection is already longer than the desired headline size. For this reason, most early headline generation work focused on either extracting and reordering  $n$ -grams from the document to be summarized (Banko et al., 2000), or extracting one or two informative sentences from the document and performing linguistically-motivated transformations to them in order to reduce the summary length (Dorr et al., 2003). The first approach is not guaranteed to produce grammatical headlines, whereas the second approach is tightly tied to the actual wording found in the document. Single-document headline generation was also explored at the Document Understanding Conferences between 2002 and 2004<sup>1</sup>.

In later years, there has been more interest in problems such as sentence compression (Galley and McKeown, 2007; Clarke and Lapata, 2008; Cohn and Lapata, 2009; Napoles et al., 2011; Berg-Kirkpatrick et al., 2011), text simplification (Zhu et al., 2010; Coster and Kauchak, 2011; Woodsend and Lapata, 2011) and sentence fusion (Barzilay and McKeown, 2005; Wan et al., 2007; Filippova and Strube, 2008; Elsner and Santhanam, 2011). All of them have direct applications for headline generation, as it can be construed as selecting one or a few sentences from the original document(s), and then reducing them to the target title size. For example, Wan et al. (2007) generate novel utterances by combining Prim’s maximum-spanning-tree algorithm with an  $n$ -gram language model to enforce fluency. Unlike HEADY, the method by Wan and colleagues is an extractive method that can summarize single documents into a sentence, as opposed to generating a sentence that can stand for a whole collec-

<sup>1</sup><http://duc.nist.gov/>

tion of news. Filippova (2010) reports a system that is very close to our settings: the input is a collection of related news articles, and the system generates a headline that describes the main event. This system uses sentence compression techniques and benefits from the redundancy in the collection. One difference with respect to HEADY is that it does not use any syntactic information aside from part-of-speech tags, and it does not require a training step. We have used this approach as a baseline for comparison.

There are not many fully abstractive systems for news summarization. The few that exist, such as the work by Genest and Lapalme (2012), rely on manually written generation templates. In contrast, HEADY automatically learns the templates or headline patterns automatically, which allows it to work in open-domain settings without relying on supervision or manual annotations.

**Open-domain pattern learning.** Pattern learning for relation extraction is an active area of research that is very related to our problem of event pattern learning for headline generation. TextRunner (Yates et al., 2007), ReVerb (Fader et al., 2011) and NELL (Carlson et al., 2010; Mohamed et al., 2011) are some examples of open-domain systems that learn surface patterns that express relations between pairs of entities. PATTY (Nakashole et al., 2012) generalizes the patterns to also include syntactic information and ontological (class membership) constraints. Our patterns are more similar to the ones used by PATTY, which also produces clusters of synonymous patterns. The main differences are that (a) HEADY is not limited to consider patterns expressing relations between pairs of entities; (b) we identify synonym patterns using a probabilistic, Bayesian approach that takes advantage of the multiplicity of news sources reporting the same events. Chambers and Jurafsky (2009) present an unsupervised method for learning narrative schemas from news, i.e., coherent sets of events that involve specific entity types (semantic roles). Similarly to them, we move from the assumptions that 1) utterances involving the same entity types within the same document (in our case, a collection of related documents) are likely describing aspects of the same event, and 2) meaningful representations of the underlying events can be learned by clustering these utterances in a principled way.

**Noisy-OR networks.** Noisy-OR Bayesian networks (Pearl, 1988) have been applied in the past to a wide class of large-scale probabilistic inference problems, from the medical domain (Middleton et al., 1991; Jaakkola and Jordan, 1999; Onisko et al., 2001), to synthetic image-decomposition and co-citation data analysis (Šingliar and Hauskrecht, 2006). By assuming independence between the causes of the hidden variables, noisy-OR models tend to be reliable (Friedman and Goldszmidt, 1996) as they require a relatively small number of parameters to be estimated (linear with the size of the network).

### 3 Headline generation

In this section, we describe the HEADY system for news headline abstraction. Our approach takes as input, for training, a corpus of news articles organized in news collections. Once the model is trained, it can generate headlines for new collections. An outline of HEADY’s main components follows (details of each component are provided in Sections 3.1, 3.2 and 3.3):

**Pattern extraction.** Identify, in each of the news collections, syntactic patterns connecting  $k$  entities, for  $k \geq 1$ . These will be the candidate patterns expressing events.

**Training.** Train a Noisy-OR Bayesian network on the co-occurrence of syntactic patterns. Each pattern extracted in the previous step is added as an observed variable, and latent variables are used to represent the hidden events that generate patterns. An additional *noise* variable links to every terminal node, allowing every terminal to be generated by language background (noise) instead of by an actual event.

**Inference.** Generate a headline from an unseen news collection. First, patterns are extracted using the pattern extraction procedure mentioned above. Given the patterns, the posterior probability of the hidden *event* variables is estimated. Then, from the activated hidden events, the likelihood of every pattern can be estimated, even if they do not appear in the collection. The single pattern with the maximum probability is selected to generate a new headline from it. Being the product of extra-news collection generalization, the retrieved pattern is more likely to be objective and informative than patterns directly observed in the news collection.

**Algorithm 1** COLLECTIONTOPATTERNS $_{\Psi}(\mathcal{N})$ :  
 $\mathcal{N}$  is a repository of news collections,  $\Psi$  is a set of parameters controlling the extraction process.

---

```

 $\mathcal{R} \leftarrow \{\}$ 
for all  $N \in \mathcal{N}$  do
  PREPROCESSDATA( $N$ )
   $E \leftarrow$  GETRELEVANTENTITIES( $N'$ )
  for all  $E_i \leftarrow$  COMBINATIONS $_{\Psi}(E)$  do
    for all  $n \in N$  do
       $\mathcal{P} \leftarrow$  EXTRACTPATTERNS $_{\Psi}(n, E_i)$ 
       $\mathcal{R}\{N, E_i\} \leftarrow \mathcal{R}\{N, E_i\} \cup \mathcal{P}$ 
return  $\mathcal{R}$ 

```

---

### 3.1 Pattern extraction

In this section we detail the process for obtaining the event patterns that constitute the building blocks of learning and inference.

Patterns are extracted from a large repository  $\mathcal{N}$  of news collections  $N_1, \dots, N_{|\mathcal{N}|}$ . Each news collection  $N = \{n_i\}$  is an unordered collection of related news, each of which can be seen as an ordered sequence of sentences, i.e.:  $n = [s_0, \dots, s_{|n|}]$ .

Algorithm 1 presents a high-level view of the pattern extraction process. The different steps are described below:

**PREPROCESSDATA:** We start by preprocessing all the news in the news collections with a standard NLP pipeline: tokenization and sentence boundary detection (Gillick, 2009), part-of-speech tagging, dependency parsing (Nivre, 2006), coreference resolution (Haghighi and Klein, 2009) and entity linking based on Wikipedia and Freebase. Using the Freebase dataset, each entity is annotated with all its Freebase types (class labels). In the end, for each entity mentioned in the document we have a unique identifier, a list with all its mentions in the document and a list of class labels from Freebase.

As a result of this process, we obtain for each sentence in the corpus a representation as exemplified in Figure 1 (1). In this example, the mentions of three distinct entities have been identified, i.e.,  $e_1, \dots, e_3$ . In the Freebase list of types (class labels),  $e_1$  is a *person* and a *celebrity*, and  $e_3$  is a *state* and a *location*.

**GETRELEVANTENTITIES:** For each news collection  $N$  we collect the set  $E$  of the entities mentioned most often within the collection. Next, we generate the set COMBINATIONS $_{\Psi}(E)$  consisting

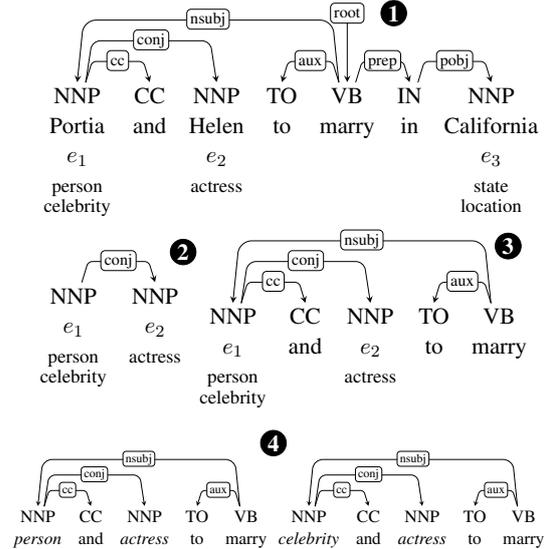


Figure 1: Pattern extraction process from an annotated dependency parse. (1): an MST is extracted from the entity pair  $e_1, e_2$  (2); nodes are heuristically added to the MST to enforce grammaticality (3); entity types are recombined to generate the final patterns (4).

of non-empty subsets of  $E$ , without repeated entities. The number of entities to consider in each collection, and the maximum size for the subsets of entities to consider are meta-parameters embedded in  $\Psi$ .<sup>2</sup>

**EXTRACTPATTERNS:** For each subset of relevant entities  $E_i$ , event patterns are mined from the articles in the news collection. The process by which patterns are extracted from a news is explained in Algorithm 2 and exemplified graphically in Figure 1 (2–4).

**GETMENTIONNODES:** Using the dependency parse  $T$  for a sentence  $s$ , we first identify the set of nodes  $M_i$  that mention the entities in  $E_i$ . If  $T$  does not contain exactly one mention of each target entity in  $E_i$ , then the sentence is ignored. Otherwise, we obtain the minimum spanning tree for the nodeset  $P_i$ , i.e., the shortest path in the dependency tree connecting all the nodes in  $M_i$  (Figure 1, 2).  $P_i$  is the set of nodes around which the patterns will be constructed.

**APPLYHEURISTICS:** With very high probability, the MST  $P_i$  that we obtain does not constitute a grammatical or useful extrapolation of the original sentence  $s$ . For example, the MST for the en-

<sup>2</sup>As our objective is to generate very short titles (under 10 words), we only consider combinations of up to three elements of  $E$ .

---

**Algorithm 2**  $\text{EXTRACTPATTERNS}_\Psi(\mathbf{n}, E_i)$ :  $\mathbf{n}$  is the list of sentences in a news article. Sentences are POS-tagged, dependency parsed and annotated with respect to a set of entities  $E \supseteq E_i$

---

```

 $\mathcal{P} \leftarrow \emptyset$ 
for all  $s \in \mathbf{n}[0 : 2)$  do
   $T \leftarrow \text{DEPPARSE}(s)$ 
   $M_i \leftarrow \text{GETMENTIONNODES}(t, E_i)$ 
  if  $\exists e \in E_i, \text{count}(e, M_i) \neq 1$  then continue
   $P_i \leftarrow \text{GETMINIMUMSPANNINGTREE}_\Psi(M_i)$ 
   $\text{APPLYHEURISTICS}_\Psi(P_i)$  or continue
   $\mathcal{P} \leftarrow \mathcal{P} \cup \text{COMBINEENTITYTYPES}_\Psi(P_i)$ 
return  $\mathcal{P}$ 

```

---

tity pair  $\langle e_1, e_2 \rangle$  in the example does not provide a good description of the event as it is neither adequate nor fluent. For this reason, we apply a set of post-processing heuristic transformations that aim at including a minimal set of meaningful nodes. These include making sure that both the root of the clause and its subject appear in the extracted pattern, and that conjunctions between entities should not be dropped (Figure 1, 3).

**COMBINEENTITYTYPES**: Finally, a distinct pattern is generated from each possible combination of entity type assignments for the participating entities. (Figure 1, 4).

It is important to note that both at training and test time, for pattern extraction we only consider the title and the first sentence of the article body. The reason is that we want to limit ourselves, in each news collection, to the most relevant event reported in the collection, which appears most of the times in these two sentences. Unlike titles, first sentences do not extensively use puns or rhetorics as they tend to be grammatical and informative rather than catchy.

The patterns mined from the same news collection and for the same set of entities are grouped together, and constitute the building blocks of the clustering algorithm which is described below.

### 3.2 Training

The extracted patterns are used to learn a Noisy-OR (Pearl, 1988) model by estimating the probability that each (observed) pattern activates one or many (hidden) *events*. Figure 2 represents the two levels: the hidden event variables at the top, and the observed pattern variables at the bottom. An additional *noise* variable links to every termi-

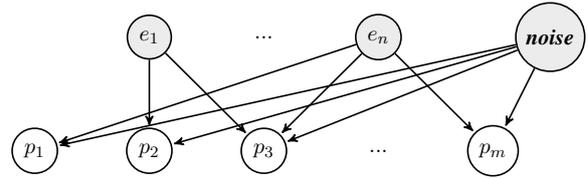


Figure 2: Probabilistic model. The associations between latent event variables and observed pattern variables are modeled by noisy-OR gates. Events are assumed to be marginally independent, and patterns conditionally independent given the events.

nal node, allowing all terminals to be generated by language background (noise) instead of by an actual event. The associations between latent events and observed patterns are modeled by noisy-OR gates.

In this model, the conditional probability of a hidden event  $e_i$  given a configuration of observed patterns  $\mathbf{p} \in \{0, 1\}^{|\mathcal{P}|}$  is calculated as:

$$\begin{aligned}
 P(e_i = 0 \mid \mathbf{p}) &= (1 - q_{i0}) \prod_{j \in \pi_j} (1 - q_{ij})^{p_j} \\
 &= \exp \left( -\theta_{i0} - \sum_{j \in \pi_i} \theta_{ij} p_j \right),
 \end{aligned}$$

where  $\pi_i$  is the set of active events (i.e.,  $\pi_i = \cup_j \{p_j\} \mid p_j = 1$ ), and  $q_{ij} = P(e_i = 1 \mid p_j = 1)$  is the estimated probability that the observed pattern  $p_i$  can, in isolation, activate the event  $e$ . The term  $q_{i0}$  is the so-called “noise” term of the model, and it accounts for the fact that an observed event  $e_i$  might be activated by some pattern that has never been observed (Jaakkola and Jordan, 1999).

In Algorithm 1, at the end of the process we group in  $\mathcal{R}[N, E_i]$  all the patterns extracted from the same news collection  $N$  and entity sub-set  $E_i$ . These groups represent rough clusters of patterns, that we can use to bootstrap the optimization of the model parameters  $\theta_{ij} = -\log(1 - q_{ij})$ . We initiate the training process by randomly selecting 100,000 of these groups, and optimize the weights of the model through 40 EM (Dempster et al., 1977) iterations.

### 3.3 Inference (generation of new headlines)

Given an unseen news collection  $N$ , the inference component of HEADY generates a single headline that captures the main event reported by the news in  $N$ . In order to do so, we first need to select a

single event-pattern  $p^*$  that is especially relevant for  $N$ . Having selected  $p^*$ , in order to generate a headline it is sufficient to replace the entity placeholders in  $p^*$  with the surface forms observed in  $N$ .

To identify  $p^*$ , we start from the assumption that the most descriptive event encoded by  $N$  must describe an important situation in which some subset of the relevant entities  $E$  in  $N$  are involved.

The basic inference algorithm is a two-step random walk in the Bayesian network. Given a set of entities  $E$  and sentences  $\mathbf{n}$ ,  $\text{EXTRACTPATTERNS}_\Psi(\mathbf{n}, E)$  collects patterns involving those entities. By normalizing the frequency of the extracted patterns, we get a probability distribution over the observed variables in the network. A two-step random walk traversing to the latent event nodes and back to the pattern nodes allows us to generalize across events. We call this algorithm  $\text{INFERENCE}(\mathbf{n}, E)$ .

In order to decide which is the most relevant set of events that should appear in the headline, we use the following procedure:

1. Given the set of entities  $E$  mentioned in the news collection, we consider each entity subset  $E_i \subseteq E$  including up to three entities<sup>3</sup>. For each  $E_i$ , we run  $\text{INFERENCE}(\mathbf{n}, E_i)$ , which computes a distribution  $\mathbf{w}_i$  over patterns involving the entities in  $E_i$ .
2. We invoke again  $\text{INFERENCE}$ , now using at the same time all the patterns extracted for every subset of  $E_i \subseteq E$ . This computes a probability distribution  $\mathbf{w}$  over all patterns involving any admissible subset of the entities mentioned in the collection.
3. Third, we select the entity-specific distribution that approximates better the overall distribution

$$\mathbf{w}^* = \arg \max_i \cos(\mathbf{w}, \mathbf{w}_i)$$

We assume that the corresponding set of entities  $E_i$  are the most central entities in the collection and therefore any headline should make sure to mention them all.

<sup>3</sup>As we noted before, we impose this limitation to keep the generated headlines relatively short and to limit data sparsity issues.

4. Finally, we select the pattern with the highest weight in  $\mathbf{w}^*$  as the pattern that better captures the main event reported in the news collection:

$$p^* = p_j \mid w_j = \arg \max_j w_j^*$$

The headline is then produced from  $p^*$ , replacing placeholders with the entities in the document from which the pattern was extracted.

While in many cases information about entity types would be sufficient to decide about the order of the entities in the generated sentences (e.g., “[person] married in [location]” for the entity set  $\{e_a = \text{“Mr. Brown”}, e_b = \text{“Los Angeles”}\}$ ), in other cases class assignment can be ambiguous (e.g., “[person] killed [person]” for  $\{e_a = \text{“Mr. A”}, e_b = \text{“Mr. B”}\}$ ). To handle these cases, when extracting patterns for an entity set  $\{e_a, e_b\}$ , we keep track of the alphabetical ordering of the entities, e.g., from a news collection about “Mr. B” killing “Mr. A” we would produce patterns such as “[person:2] killed [person:1]” or “[person:1] was killed by [person:2]” since  $e_a = \text{“Mr. A”} < e_b = \text{“Mr. B”}$ . At inference time, when we query the model with such patterns we can only activate events whose assignments are compatible with the entities observed in the text, making the replacement straightforward and unambiguous.

## 4 Experiment settings

In our method we use patterns that are fully lexicalized (with the exception of entity placeholders) and enriched with syntactic data. Under these circumstances, the Noisy-OR can effectively generalize and learn meaningful clusters only if provided with large amounts of data. To our best knowledge, available data sets for headline generation are not large enough to support this kind of inference.

For this reason, we rely on a corpus of news crawled from the web between 2008 and 2012 which have been clustered based on closeness in time and cosine similarity, using the vector-space model and tf.idf weights. News collections with less than 5 documents are discarded<sup>4</sup>, and those

<sup>4</sup>There is a very long tail of singleton articles, which do not offer useful examples of lexical or syntactic variation, and many very small collections that tend to be especially noisy, hence the decision to consider only collections with at least 5 documents.

larger than 50 documents are capped, by randomly picking 50 documents from the collection<sup>5</sup>. The total number of news collections after clustering is 1.7 million. From this set, we have set aside a few hundred collections that will remain unseen until the final evaluation.

As we have no development set, we have done no tuning of the parameters for pattern extraction nor for the Bayesian network training (100,000 latent variables to represent the different events, 40 EM iterations, as mentioned in Section 3.2). The EM iterations on the noisy-OR were distributed across 30 machines with 16 GB of memory each.

#### 4.1 Systems used

One of the questions we wanted to answer in this research was whether it was possible to obtain the same quality with automatically abstracted headlines as with human-generated headlines. For every news collection we have as many human-generated headlines as documents. To decide which human-generated headline should be used in this comparison, we used three different methods that pick one of the collection headlines:

- **Latest headline:** selects the headline from the latest document in the collection. Intuitively this should be the most relevant one for news about sport matches and competitions, where the earlier headlines offer previews and predictions, and the later headlines report who won and the final scores.
- **Most frequent headline:** some headlines are repeated across the collection, and this method chooses the most frequent one. If there are several with the same frequency, one is taken at random<sup>6</sup>.
- **TopicSum:** we use TopicSum (Haghighi and Vanderwende, 2009), a 3-layer hierarchical topic model, to infer the language model that is most central for the collection. The news title that has the smallest Kullback-Leibler

<sup>5</sup>Even though we did not run any experiment to find an optimal value for this parameter, 50 documents seems like a reasonable choice to avoid redundancy while allowing for considerable lexical and syntactic variation.

<sup>6</sup>The most frequent headline only has a tie in 6 collections in the whole test set. In 5 cases two headlines are tied at frequencies around 4, and in one case three headlines are tied at frequency 2. All six are large collections with 50 news articles, so this baseline is significantly different from a random baseline.

	R-1	R-2	R-SU4
HEADY	0.3565	0.1903	0.1966
Most frequent pattern	0.3560	0.1864	0.1959
TopicSum	0.3594	0.1821	0.1935
MSC	0.3470	0.1765	0.1855
Most frequent headline	0.3177	0.1401	0.1668
Latest headline	0.2814	0.1191	0.1425

Table 2: Results from the automatic evaluation, sorted according to the ROUGE-2 and ROUGE-SU4 scores.

divergence with respect the collection language model is the one chosen.

A headline generation system that addresses the same application as ours is (Filippova, 2010), which generates a graph from the collection sentences and selects the shortest path between the begin and the end node traversing words in the same order in which they were found in the original documents. We have used this system, called Multi-Sentence Compression (**MSC**), for comparisons.

Finally, in order to understand whether the noisy-OR Bayesian network is useful for generalizing across patterns into *latent events*, we added a baseline that extracts all patterns from the test collection following the same COLLECTIONTOPATTERNS algorithm (including the application of the linguistically motivated heuristics), and then produces a headline straightaway from the most frequent pattern extracted. In other words, the only difference with respect to HEADY is that in this case no generalization through the Noisy-OR network is carried out, and that headlines are generated from patterns directly observed in the test news collections. We call this system **Most frequent pattern**.

#### 4.2 Annotation activities

In order to evaluate HEADY’s performance, we carried out two annotation activities.

First, from the set of collections that we had set aside at the beginning, we randomly chose 50 collections for which all the systems could generate an output, and we asked raters to manually write titles for them. As this is not a simple task to be crowdsourced, for this evaluation we relied on eight trained raters. We collected between four and five reference titles for each of the fifty news collections, to be used to compare the headline

	Readability	Informativeness
TopicSum	<b>4.86</b>	<b>4.63</b>
Most freq. headline	†‡4.61	†‡◊4.43
Latest headline	†‡4.55	†4.00
HEADY	†4.28	†3.75
Most freq. pattern	†3.95	†3.82
MSC	3.00	3.05

Table 3: Results from the manual evaluation. At 95% confidence, TopicSum is significantly better than all others for readability, and only indistinguishable from the most frequent pattern for informativeness. For the rest, ◊ means being significantly better than HEADY, ‡ than the most frequent pattern, and † than MSC.

generation methods using automatic summarization metrics.

Then, we took the output of the systems for the 50 test collections and asked human raters to evaluate the headlines:

1. Raters were shown one headline and asked to rate it in terms of **readability** on a 5-point Likert scale. In the instructions, the raters were provided with examples of ungrammatical and grammatical titles to guide them in this annotation.
2. After the previous rating is done, raters were shown a selection of five documents from the collection, and they were asked to judge the **informativeness** of the previous headline for the news in the collection, again on a 5-point Likert scale.

This second annotation was carried out by independent raters in a crowd-sourcing setting. The raters did not have any involvement with the inception of the model or the writing of the paper. They did not know that the headlines they were rating were generated according to different methods. We measured inter-judge agreement on the Likert-scale annotations using their Intra-Class Correlation (ICC) (Cicchetti, 1994). The ICC for readability is 0.76 (0.95 confidence interval [0.71, 0.80]), and for informativeness it is 0.67 (0.95 confidence interval [0.60, 0.73]). This means strong agreement for readability, and moderate agreement for informativeness.

## 5 Results

The COLLECTIONTOPATTERNS algorithm was run on the training set, producing a 230 million

event patterns. Patterns that were obtained from the same collection and involving the same entities were grouped together, for a total of 1.7 million pattern collections. The pattern groups are used to bootstrap the Noisy-OR model training. Training the HEADY model that we used for the evaluation took around six hours on 30 cores.

Table 2 shows the results of the comparison of the headline generation systems using ROUGE (R-1, R-2 and R-SU4) (Lin, 2004) with the collected references. According to Owczarzak et al. (2012), ROUGE is still a competitive metric that correlates well with human judgements for ranking summarizers. The significance tests for ROUGE are performed using bootstrap resampling and a graphical significance test (Minka, 2002). The human annotators that created the references for this evaluation were explicitly instructed to write objective titles, which is the kind of headlines that the abstractive systems aim at generating. It is common to see real headlines that are catchy, joking, or with a double meaning, and therefore they use a different vocabulary than objective titles that simply mention what happened. TopicSum sometimes selects objective titles amongst the human-made titles and that is why it also scores very well with the ROUGE scores. But the other two criteria for choosing human-made headlines select non-objective titles much more often, and this lowers their performance when measured with ROUGE with respect to the objective references.

Table 3 lists the results of the manual evaluation of readability and informativeness of the generated headlines. The first result that we can see is the difference in the rankings between the two evaluations. Part of this difference might be due to the fact that ROUGE is not as good for discriminating between human-made and automatic summaries. In fact, in the DUC competitions, the gap between human summaries and automatic summaries was also more apparent in the manual evaluations than using ROUGE. Another part of the observed difference may be due to the design of the evaluation. The manual evaluation is asking raters to judge whether real, human-written titles that were actually used for those news are grammatical and informative. As could be expected, as these are published titles, the real titles score very good on the manual evaluation.

Some other interesting results are:

Model	Generated title
TopicSum	Modern Family's Eric Stonestreet laughs off Charlize Theron rumours
MSC	Modern Family star Eric Stonestreet is dating Charlize Theron.
Latest headline	Eric laughs off Theron dating rumours
Frequent pattern	Eric Stonestreet jokes about Charlize relationship
Frequent headline	Charlize Theron dating Modern Family star
HEADY	Eric Stonestreet not dating Charlize Theron
TopicSum	McFadzean rescues point for Crawley Town
MSC	Crawley side challenging for a point against Oldham Athletic.
Latest headline	Reds midfielder victim of racist tweet
Frequent pattern	Kyle McFadzean fired a equaliser Crawley were made
Frequent headline	Latics halt Crawley charge
HEADY	Kyle McFadzean rescues point for Crawley Town F.C.
TopicSum	UCI to strip Lance Armstrong of his 7 Tour titles
MSC	The international cycling union said today.
Latest headline	Letters: elderly drivers and Lance Armstrong
Frequent pattern	Lance Armstrong stripped of Tour de France titles
Frequent headline	Today in the news: third debate is tonight
HEADY	Lance Armstrong was stripped of Tour de France titles

Table 4: A comparison of the titles generated by the different models for three news collections.

- Amongst the automatic systems, HEADY performed better than MSC, with statistical significance at 95% for all the metrics. Headlines based on the most frequent patterns were better than MSC for all metrics but ROUGE-2.
- The most frequent pattern baseline and HEADY have comparable performance across all the metrics (not statistically significantly different), although HEADY has slightly better scores for all metrics except for informativeness.

While we do not take any step to explicitly model stylistic variation, estimating the weights of the Noisy-OR network turns out to be a very effective way of filtering out sensational wording to the advantage of plainer, more objective style. This may not clearly emerge from the evaluation, as we did not explicitly ask the raters to annotate the items based on their objectivity, but a manual inspection of the clusters suggests that the generalization is working in the right direction.

Table 4 presents a selection of outputs produced by the six models for three different news collections. The first example shows a news collection containing news about a rumour that was immediately denied. In the second example, HEADY generalization improves over the most frequent pattern. In the third case, HEADY generates a

good title from a noisy collection (containing different but related events). The examples also show that TopicSum is very effective in selecting a good human-generated headline for each collection. This opens the possibility of using TopicSum to automatically generate ROUGE references for future evaluations of abstractive methods.

## 6 Conclusions

We have presented HEADY, an abstractive headline generation system based on the generalization of syntactic patterns by means of a Noisy-OR Bayesian network. We evaluated the model both automatically and through human annotations. HEADY performs significantly better than a state-of-the-art open domain abstractive model (Filippova, 2010) in all evaluations, and is in par with human-generated headlines in terms of ROUGE scores. We have shown that it is possible to achieve high quality generation of news headlines in an open-domain, unsupervised setting by successfully exploiting syntactic and ontological information. The system relies on a standard NLP pipeline, requires no manual data annotation and can effectively scale to web-sized corpora.

For feature work, we plan to improve all components of HEADY in order to fill in the gap with the human-generated titles in terms of readability and informativeness. One of the directions in which we plan to move is the removal of the syntactic heuristics that currently enforce pattern well-formedness and to automatically learn the necessary transformations from the data.

Two other lines of work that we plan to explore are the possibility of personalizing the headlines to user interests (as stored in user profiles or expressed as user queries), and to investigate further applications of the Bayesian network of event patterns, such as its use for relation extraction and knowledge base population.

## Acknowledgments

The research leading to these results has received funding from: the EU's 7<sup>th</sup> Framework Programme (FP7/2007-2013) under grant agreement number 257790; the Spanish Ministry of Science and Innovation's project Holopedia (TIN2010-21128-C02); and the Regional Government of Madrid's MA2VICMR (S2009/TIC1542). We would like to thank Katja Filippova and the anonymous reviewers for their insightful comments.

## References

- Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, ACL '00, pages 318–325. Association for Computational Linguistics.
- Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.
- Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 481–490. Association for Computational Linguistics.
- Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*, pages 3–3.
- Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised Learning of Narrative Schemas and Their Participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, pages 602–610.
- Domenic V Cicchetti. 1994. Guidelines, criteria, and rules of thumb for evaluating normed and standardized assessment instruments in psychology. *Psychological Assessment*, 6(4):284.
- James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31(1):399–429.
- Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.
- William Coster and David Kauchak. 2011. Learning to simplify sentences using Wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 1–9. Association for Computational Linguistics.
- Arthur P. Dempster, Nan M. Laird, and Donald B. Rubi. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.
- Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*, pages 1–8. Association for Computational Linguistics.
- Micha Elsner and Deepak Santhanam. 2011. Learning to fuse disparate sentences. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 54–63. Association for Computational Linguistics.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics.
- Katja Filippova and Michael Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 177–185. Association for Computational Linguistics.
- Katja Filippova. 2010. Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. Association for Computational Linguistics.
- Nir Friedman and Moises Goldszmidt. 1996. Learning Bayesian networks with local structure. In *Proceedings of the Twelfth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 252–262, San Francisco, CA. Morgan Kaufmann.
- Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 180–187.
- Pierre-Etienne Genest and Guy Lapalme. 2012. Fully abstractive approach to guided summarization. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, short papers*. Association for Computational Linguistics.
- Dan Gillick. 2009. Sentence boundary detection and the problem with the us. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244. Association for Computational Linguistics.
- Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1152–1161. Association for Computational Linguistics.
- Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics.

- Tommi S. Jaakkola and Michael I. Jordan. 1999. Variational probabilistic inference and the QMR-DT Network. *Journal of Artificial Intelligence Research*, 10:291–322.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81.
- Blackford Middleton, Michael Shwe, David Heckerman, Max Henrion, Eric Horvitz, Harold Lehmann, and Gregory Cooper. 1991. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base. I. The probabilistic model and inference algorithms. *Methods of information in medicine*, 30(4):241–255, October.
- Tom Minka. 2002. Judging Significance from Error Bars. *CM U Tech R report*.
- Thahir P Mohamed, Estevam R Hruschka Jr, and Tom M Mitchell. 2011. Discovering relations between noun categories. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1447–1455. Association for Computational Linguistics.
- Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Patty: A taxonomy of relational patterns with semantic types. *EMNLP12*.
- Courtney Napoles, Chris Callison-Burch, Juri Ganitkevitch, and Benjamin Van Durme. 2011. Paraphrastic sentence compression with a character-based metric: Tightening without deletion. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 84–90. Association for Computational Linguistics.
- Joakim Nivre. 2006. *Inductive Dependency Parsing*, volume 34 of *Text, Speech and Language Technology*. Springer.
- Agnieszka Onisko, Marek J. Druzdzel, and Hanna Wasyluk. 2001. Learning Bayesian network parameters from small data sets: application of Noisy-OR gates. *International Journal of Approximated Reasoning*, 27(2):165–182.
- Karolina Owczarzak, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of the NAACL-HLT 2012 Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, pages 1–9. Association for Computational Linguistics.
- Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.
- Tomáš Šingliar and Miloš Hauskrecht. 2006. Noisy-or component analysis and its application to link analysis. *J. Mach. Learn. Res.*, 7:2189–2213, December.
- Stephen Wan, Robert Dale, Mark Dras, and Cécile Paris. 2007. Global Revision in Summarisation: Generating Novel Sentences with Prim’s Algorithm. In *Proceedings of PACLING 2007 - 10th Conference of the Pacific Association for Computational Linguistics*.
- Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 409–420. Association for Computational Linguistics.
- Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. TextRunner: Open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics.
- Zhemina Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of The 23rd International Conference on Computational Linguistics*, pages 1353–1361.

# Conditional Random Fields for Responsive Surface Realisation using Global Features

Nina Dethlefs, Helen Hastie, Heriberto Cuayahuitl and Oliver Lemon

Mathematical and Computer Sciences

Heriot-Watt University, Edinburgh

n.s.dethlefs | h.hastie | h.cuayahuitl | o.lemon@hw.ac.uk

## Abstract

Surface realisers in spoken dialogue systems need to be more responsive than conventional surface realisers. They need to be sensitive to the utterance context as well as robust to partial or changing generator inputs. We formulate surface realisation as a sequence labelling task and combine the use of conditional random fields (CRFs) with semantic trees. Due to their extended notion of context, CRFs are able to take the global utterance context into account and are less constrained by local features than other realisers. This leads to more natural and less repetitive surface realisation. It also allows generation from partial and modified inputs and is therefore applicable to incremental surface realisation. Results from a human rating study confirm that users are sensitive to this extended notion of context and assign ratings that are significantly higher (up to 14%) than those for taking only local context into account.

## 1 Introduction

Surface realisation typically aims to produce output that is grammatically well-formed, natural and cohesive. Cohesion can be characterised by lexical or syntactic cues such as repetitions, substitutions, ellipses, or connectives. In automatic language generation, such properties can sometimes be difficult to model, because they require rich context-awareness that keeps track of all (or much) of what was generated before, i.e. a growing generation history. In text generation, cohesion can span over the entire text. In interactive settings such as generation within a spoken dialogue system (SDS), a

challenge is often to keep track of cohesion over several utterances. In addition, since interactions are dynamic, generator inputs from the dialogue manager can sometimes be partial or subject to subsequent modification. This has been addressed by work on incremental processing (Schlangen and Skantze, 2009). Since dialogue acts are passed on to the generation module as soon as possible, this can sometimes lead to incomplete generator inputs (because the user is still speaking), or inputs that are subject to later modification (because of an initial ASR mis-recognition).

In this paper, we propose to formulate surface realisation as a sequence labelling task. We use conditional random fields (Lafferty et al., 2001; Sutton and McCallum, 2006), which are suitable for modelling rich contexts, in combination with semantic trees for rich linguistic information. This combination is able to keep track of dependencies between syntactic, semantic and lexical features across multiple utterances. Our model can be trained from minimally labelled data, which reduces development time and may (in the future) facilitate an application to new domains.

The domain used in this paper is a pedestrian walking around a city looking for information and recommendations for local restaurants from an SDS. We describe here the module for surface realisation. Our main hypothesis is that the use of global context in a CRF with semantic trees can lead to surface realisations that are better phrased, more natural and less repetitive than taking only local features into account. Results from a human rating study confirm this hypothesis. In addition, we compare our system with alternative surface realisation methods from the literature, namely, a rank and boost approach and  $n$ -grams.

Finally, we argue that our approach lends itself

to surface realisation within incremental systems, because CRFs are able to model context across full as well as partial generator inputs which may undergo modifications during generation. As a demonstration, we apply our model to incremental surface realisation in a proof-of-concept study.

## 2 Related Work

Our approach is most closely related to Lu et al. (2009) who also use CRFs to find the best surface realisation from a semantic tree. They conclude from an automatic evaluation that using CRF-based generation which takes long-range dependencies into account outperforms several baselines. However, Lu et al.'s generator does not take context beyond the current utterance into account and is thus restricted to local features. Furthermore, their model is not able to modify generation results on the fly due to new or updated inputs.

In terms of surface realisation from graphical models (and within the context of SDSs), our approach is also related to work by Georgila et al. (2002) and Dethlefs and Cuayáhuitl (2011b), who use HMMs, Dethlefs and Cuayáhuitl (2011a) who use Bayes Nets, and Mairesse et al. (2010) who use Dynamic Bayes Nets within an Active Learning framework. The last approach is also concerned with generating restaurant recommendations within an SDS. Specifically, their system optimises its performance online, during the interaction, by asking users to provide it with new textual descriptions of concepts, for which it is unsure of the best realisation. In contrast to these related approaches, we use undirected graphical models which are useful when the natural directionality between the input variables is unknown.

In terms of surface realisation for SDSs, Oh and Rudnicky (2000) present foundational work in using an  $n$ -gram-based system. They train a surface realiser based on a domain-dependent language model and use an overgeneration and ranking approach. Candidate utterances are ranked according to a penalty function which penalises too long or short utterances, repetitious utterances and utterances which either contain more or less information than required by the dialogue act. While their approach is fast to execute, it has the disadvantage of not being able to model long-range dependencies. They show that humans rank their output equivalently to template-based generation.

Further, our approach is related to the SPaRKY

sentence generator (Walker et al., 2007). SPaRKY was also developed for the domain of restaurant recommendations and was shown to be equivalent to or better than a carefully designed template-based generator which had received high human ratings in the past (Stent et al., 2002). It generates sentences in two steps. First, it produces a randomised set of alternative realisations, which are then ranked according to a mapping from sentence plans to predicted human ratings using a boosting algorithm. As in our approach, SPaRKY distinguishes local and global features. Local features take only information of the current tree node into account, including its parents, siblings and children, while global features take information of the entire utterance into account. While SPaRKY is shown to reach high output quality in comparison to a template-based baseline, the authors acknowledge that generation with SPaRKY is rather slow when applied in a real-time SDS. This could present a problem in incremental settings, where generation speed is of particular importance.

The SPaRKY system is also used by Rieser et al. (2011), who focus on information presentation strategies for restaurant recommendations, summaries or comparisons within an SDS. Their surface realiser is informed by the highest ranked SPaRKY outputs for a particular information presentation strategy and will constitute one of our baselines in the evaluation.

More work on trainable realisation for SDSs generally includes Bulyko and Ostendorf (2002) who use finite state transducers, Nakatsu and White (2006) who use supervised learning, Varges (2006) who uses chart generation, and Konstas and Lapata (2012) who use weighted hypergraphs, among others.

## 3 Cohesion across Utterances

### 3.1 Tree-based Semantic Representations

The restaurant recommendations we generate can include any of the attributes shown in Table 1. It is then the task of the surface realiser to find the best realisation, including whether to present them in one or several sentences. This often is a sentence planning decision, but in our approach it is handled using CRF-based surface realisation. The semantic forms underlying surface realisation can be produced in many ways. In our case, they are produced by a reinforcement learning agent which orders semantic attributes in the tree ac-

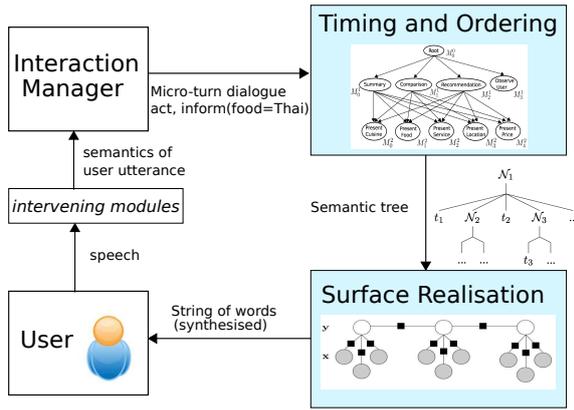


Figure 1: Architecture of our SDS with a focus on the NLG components. While the user is speaking, the dialogue manager sends dialogue acts to the NLG module, which uses reinforcement learning to order semantic attributes and produce a semantic tree (see Dethlefs et al. (2012b)). This paper focuses on surface realisation from these trees using a CRF as shown in the surface realisation module.

Slot	Example
ADDRESS	The venue's address is ...
AREA	It is located in ...
FOOD	The restaurant serves ... cuisine.
NAME	The restaurant's name is ...
PHONE	The venue's phone number is ...
POSTCODE	The postcode is ...
QUALITY	This is a ... venue.
PRICE	It is located in the ... price range.
SIGNATURE	The venue specialises in ...
VENUE	This venue is a ...

Table 1: Semantic slots required for our domain along with example realisations. Attributes can be combined in all possible ways during generation.

according to their confidence in the dialogue. This is because SDSs can often have uncertainties with regard to the user's actual desired attribute values due to speech recognition inaccuracies. We therefore model all semantic slots as probability distributions, such as  $inform(food=Indian, 0.6)$  or  $inform(food=Italian, 0.4)$  and apply reinforcement learning to finding the optimal sequence for presentation. Please see Dethlefs et al. (2012b) for details. Here, we simply assume that a semantic form has been produced by a previous processing module.

As shown in the architecture diagram in Figure 1, a CRF surface realiser takes a semantic tree as input. We represent these as context-free trees which can be defined formally as 4-tuples

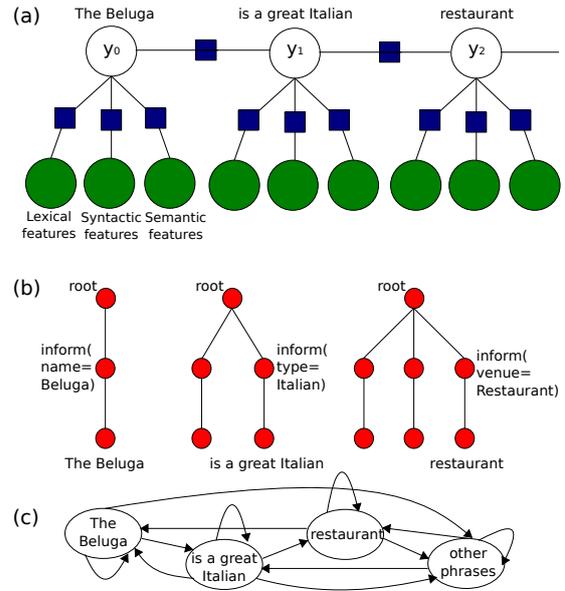


Figure 2: (a) Graphical representation of a linear-chain Conditional Random Field (CRF), where empty nodes correspond to the labelled sequence, shaded nodes to linguistic observations, and dark squares to feature functions between states and observations; (b) Example semantic trees that are updated at each time step in order to provide linguistic features to the CRF (only one possible surface realisation is shown and parse categories are omitted for brevity); (c) Finite state machine of phrases (labels) for this example.

$\{S, T, N, H\}$ , where  $S$  is a start symbol, typically the root node of the tree;  $T = \{t_0, t_1, t_2 \dots t_{|T|}\}$  is a set of terminal symbols, corresponding to single phrases;  $N = \{n_0, n_1, n_2 \dots n_{|N|}\}$  is a set of non-terminal symbols corresponding to semantic categories, and  $H = \{h_0, h_1, h_2 \dots h_{|H|}\}$  is a set of production rules of the form  $n \rightarrow \alpha$ , where  $n \in N$ ,  $\alpha \in T \cup N$ . The production rules represent alternatives at each branching node where the CRF is consulted for the best available expansion from the subset of possible ones. All nodes in the tree are annotated with a semantic concept (obtained from the semantic form) as well as their parse category.

### 3.2 Conditional Random Fields for Phrase-Based Surface Realisation

The main idea of our approach is to treat surface realisation as a sequence labelling task in which a sequence of semantic inputs needs to be labelled with appropriate surface realisations. The task is therefore to find a mapping between (observed)

lexical, syntactic and semantic features and a (hidden) best surface realisation.

We use the linear-chain Conditional Random Field (CRF) model for statistical phrase-based surface realisation, see Figure 2 (a). This probabilistic model defines the posterior probability of labels (surface realisation phrases)  $\mathbf{y}=\{y_1, \dots, y_{|\mathbf{y}|}\}$  given features  $\mathbf{x}=\{x_1, \dots, x_{|\mathbf{x}|}\}$  (informed by a semantic tree, see Figure 2 (b)), as

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^T \exp \left\{ \sum_{k=1}^K \theta_k \Phi_k(y_t, y_{t-1}, \mathbf{x}_t) \right\},$$

where  $Z(\mathbf{x})$  is a normalisation factor over all possible realisations (i.e. labellings) of  $\mathbf{x}$  such that the sum of all terms is one. The parameters  $\theta_k$  are weights corresponding to feature functions  $\Phi_k(\cdot)$ , which are real values describing the label state  $y$  at time  $t$  based on the previous label state  $y_{t-1}$  and features  $\mathbf{x}_t$ . For example: from Figure 2 (c),  $\Phi_k$  might have the value  $\Phi_k = 1.0$  for the transition from “*The Beluga*” to “*is a great Italian*”, and 0.0 elsewhere. The parameters  $\theta_k$  are set to maximise the conditional likelihood of phrase sequences in the training data set. They are estimated using the gradient ascent algorithm.

After training, labels can be predicted for new sequences of observations. The most likely phrase sequence is expressed as

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}),$$

which is computed using the Viterbi algorithm. We use the Mallet package<sup>1</sup> (McCallum, 2002) for parameter learning and inference.

### 3.3 Feature Selection and Training

The following features define the generation context used during training of the CRF. The generation context includes everything that has been generated for the current utterance so far. All features can be obtained from a semantic input tree.

- Lexical items of parents and siblings,
- Semantic types in expansion,
- Semantic types of parents and siblings,
- Parse category of expansion,
- Parse categories of parents and siblings.

We use the StanfordParser<sup>2</sup> (Marneffe et al., 2006) to obtain the parse category for each tree node.

<sup>1</sup><http://mallet.cs.umass.edu/>

<sup>2</sup><http://nlp.stanford.edu/software/lex-parser.shtml>

The semantics for each node are derived from the input dialogue acts (these are listed in Table 1) and are associated with nodes. The lexical items are present in the generation context and are mapped to semantic tree nodes.

As an example, for generating an utterance (label sequence) such as *The Beluga is a great restaurant. It is located in the city centre.*, each generation step needs to take the features of the entire generation history into account. This includes all individual lexical items generated, the semantic types used and the parse categories for each tree node involved. For the first constituent, *The Beluga*, this corresponds to the features  $\{\hat{\text{BEGIN NAME}}\}$  indicating the beginning of a sentence (where empty features are omitted), the beginning of a new generation context and the next semantic slot required. For the second constituent, *is a great restaurant*, the features are  $\{\text{THE BELUGA NAME NP VENUE}\}$ , i.e. including the generation history (with lexical items and parse category added for the first constituent) and the semantics of the next required slot, VENUE. In this way, a sequence of surface form constituents is generated corresponding to latent states in the CRF.

Since global utterance features capture the full generation context (i.e. beyond the current utterance), we are also able to model phenomena such as co-references and pronouns. This is useful for longer restaurant recommendations which may span over more than one utterance. If the generation history already contains a semantic attribute, e.g. the restaurant name, the CRF may afterwards choose a pronoun, e.g. *it*, which has a higher likelihood than using the proper name again. Similarly, the CRF may decide to realise a new attribute as constituents of different order, such as a sentence or PP, depending on the length, number and parse categories of previously generated output. In this way, our approach implicitly treats sentence planning decisions such as the distribution of content over a set of messages in the same way as (or as part of) surface realisation. A further capability of our surface realiser is that it can generate complete phrases from full as well as partial dialogue acts. This is useful in interactive contexts, where we need as much robustness as possible. A demonstration of this is given in Section 5 in an application to incremental surface realisation.

To train the CRF, we used a data set of 552 restaurant recommendations from the website The

List.<sup>3</sup> The data contains recommendations such as *Located in the city centre, Beluga is a stylish yet laid-back restaurant with a smart menu of modern European cuisine.*

### 3.4 Grammar Induction

The grammar  $g$  of surface realisation candidates is obtained through an automatic grammar induction algorithm which can be run on unlabelled data and requires only minimal human intervention. This grammar defines the surface realisation space for the CRFs. We provide the human corpus of restaurant recommendations from Section 3.3 as input to grammar induction. The algorithm is shown in Algorithm 1. It first identifies all semantic attributes of interest in an utterance, in our case those specified in Table 1, and replaces them by a variable. These attributes include food types, such as *Mexican*, *Chinese*, particular parts of town, prices, etc. About 45% of them can be identified based on heuristics. The remainder needs to be hand-annotated at the moment, which includes mainly attributes like restaurant names or quality attributes, such as *delicate*, *exquisite*, etc. Subsequently, all utterances are parsed using the Stanford parser to obtain constituents and are integrated into the grammar under construction. The non-terminal symbols are named after the automatically annotated semantic attributes contained in their expansion, e.g.  $\text{NAME\_QUALITY} \rightarrow \textit{The } \$name\$ \textit{ is of } \$quality\$ \textit{ quality}$ . In this way, each non-terminal symbol has a semantic representation and an associated parse category. In total, our induced grammar contains more than 800 rules.

## 4 Evaluation

To evaluate our approach, we focus on a subjective human rating study which aims to determine whether CRF-based surface realisation that takes the full generation context into account, called **CRF (global)**, is perceived better by human judges than one that uses a CRF but just takes local context into account, called **CRF (local)**. While CRF (global) uses features from the entire generation history, CRF (local) uses only features from the current tree branch. We assume that cohesion can be identified by untrained judges as natural, well-phrased and non-repetitive surface forms. To examine differences in methodology between

<sup>3</sup><http://www.list.co.uk>

---

### Algorithm 1 Grammar Induction.

---

```
1: function FINDGRAMMAR(utterances  $u$ , semantic attributes  $a$ ) return grammar
2:   for each utterance  $u$  do
3:     if  $u$  contains a semantic attribute from  $a$ , such as
       venue, cuisine, etc. then
4:       Find and replace the attribute by its semantic
       variable, e.g.  $\$venue\$$ .
5:     end if
6:     Parse the sentence and induce a set of rules  $\alpha \rightarrow \beta$ ,
       where  $\alpha$  is a semantic variable and  $\beta$  is its parse.
7:     Traverse the parse tree in a top-down, depth-first
       search and
8:     if expansion  $\beta$  exists then
9:       continue
10:    else if non-terminal  $\alpha$  exists then
11:      add new expansion  $\beta$  to  $\alpha$ .
12:    else write new rule  $\alpha \rightarrow \beta$ .
13:    end if
14:    Write grammar.
15:  end for
16: end function
```

---

CRFs and other state-of-the-art methods, we also compare our system to two other baselines:

- **CLASSiC** corresponds to the system reported in Rieser et al. (2011),<sup>4</sup> which generates restaurant recommendations based on the SPaRKY system (Walker et al., 2007), and has received high ratings in the past. SPaRKY uses global utterance features.
- **$n$ -grams** represents a simple 5-gram baseline that is similar to Oh and Rudnicky (2000)’s system. We will sample from the most likely slot realisations that do not contain a repetition and include exactly the required slot values. Local context only is taken into account.

### 4.1 Human Rating Study

We carried out a user rating study on the CrowdFlower crowd sourcing platform.<sup>5</sup> Each participant was shown part of a real human-system dialogue that emerged as part of the CLASSiC project evaluation (Rieser et al., 2011). All dialogues and data are freely available from <http://www.classic-project.org>. Each dialogue contained two variations for one of the utterances. These variations were generated from two out of the four systems described above. The order that these were presented to the participant was counterbalanced. Table 2 gives an example of a dialogue segment presented to the participants.

<sup>4</sup>In Rieser et al. (2011), this system is referred to as the TIP system, which generates summaries, comparisons or recommendations for restaurants. For the present study, we com-

**SYS** Thank you for calling the Cambridge Information system. Your call will be recorded for research purposes. You may ask for information about a place to eat, such as a restaurant, a pub, or a cafe. How may I help you?

**USR** I want to find an American restaurant which is in the very expensive area.

**SYS A** *The restaurant Gourmet Burger is an outstanding, expensive restaurant located in the central area.*

**SYS B** *Gourmet Burger is a smart and welcoming restaurant. Gourmet Burger provides an expensive dining experience with great food and friendly service. If you're looking for a central meal at an expensive price.*

**USR** What is the address and phone number?

**SYS** Gourmet Burger is on Regent Street and its phone number is 01223 312598.

**USR** Thank you. Good bye.

Table 2: Example dialogue for participants to compare alternative outputs in italics, USR=user, SYS A=CRF (global), SYS B=CRF(local).

System	Natural	Phrasing	Repetit.
CRF global	<b>3.65</b>	<b>3.64</b>	<b>3.65</b>
CRF local	3.10*	3.19*	3.13*
CLASSiC	3.53*	3.59	3.48*
<i>n</i> -grams	3.01*	3.09*	3.32*

Table 3: Subjective user ratings. Significance with CRF (global) at  $p < 0.05$  is indicated as \*.

44 participants gave a total of 1,830 ratings of utterances produced across the four systems. Fluent speakers of English only were requested and the participants were from the United States. They were asked to rate each utterance on a 5 point Likert scale in response to the following questions (where 5 corresponds to *totally agree* and 1 corresponds to *totally disagree*):

- The utterance was natural, i.e. it could have been produced by a human. (*Natural*)
- The utterance was phrased well. (*Phrasing*)
- The utterance was repetitive. (*Repetitive*)

## 4.2 Results

We can see from Table 3 that across all the categories, the CRF (global) gets the highest overall ratings. This difference is significant for all categories compared with CRF (local) and *n*-grams (using a 1-sided Mann Whitney U-test,  $p < 0.001$ ).

pare only with the subset of recommendations.

<sup>5</sup><http://www.crowdfunder.com>

Possibly this is because the local context taken into account by both systems was not enough to ensure cohesion across surface phrases. It is not possible, e.g., to cover co-references within a local context only or discourse markers that refer beyond the current utterance. This can lead to short and repetitive phrases, such as *Make your way to Gourmet Burger. The food quality is outstanding. The prices are expensive.* generated by the *n*-gram baseline.

The CLASSiC baseline, based on SPaRKY, was the most competitive system in our comparison. None-the-less CRF (global) is rated higher across categories and significantly so for *Natural* ( $p < 0.05$ ) and *Repetitive* ( $p < 0.005$ ). For *Phrasing*, there is a trend but not a significant difference ( $p < 0.16$ ). All comparisons are based on a 1-sided Mann Whitney U-test. A qualitative comparison between the CRF (global) and CLASSiC outputs showed the following. CLASSiC utterances tend to be longer and contain more sentences than CRF (global) utterances. While CRF (global) often decides to aggregate attributes into one sentence, such as *the Beluga is an outstanding restaurant in the city centre*, CLASSiC tends to rely more on individual messages, such as *The Beluga is an outstanding restaurant. It is located in the city centre.* A possible reason is that while CRF (global) is able to take features beyond an utterance into account, CLASSiC/SPaRKY is restricted to global features of the current utterance.

We can further compare our results with Rieser et al. (2011) and Mairesse et al. (2010) who also generate restaurant recommendations and asked similar questions to participants as we did. Rieser et al. (2011)'s system received an average rating of 3.58<sup>6</sup> in terms of *Phrasing* which compares to our 3.64. This difference is not significant, and in line with the user ratings we observed for the CLASSiC system above (3.59). Mairesse et al. (2010) achieved an average score of 4.05 in terms of *Natural* in comparison to our 3.65. This difference is significant at  $p < 0.05$ . Possibly their better performance is due to the data set being more "in domain" than ours. They collected data from humans that was written specifically for the task that the system was tested on. In contrast, our system was trained on freely available data that was written by professional restaurant reviewers. Unfortunately, we cannot compare across other categories,

<sup>6</sup>This was rescaled from a 1-6 scale.

USR1 I'm looking for a nice restaurant in the centre.  
 SYS1 *inform(area=centre [0.2], food=Thai [0.3])*  
*inform(name=Bangkok [0.3])*  
 So you're looking for a Thai ...  
 USR2 [*bares in*] No, I'm looking for a restaurant  
 with good quality food.  
 SYS2 *inform(quality=good [0.6], name=Beluga [0.6])*  
 Oh sorry, so a nice restaurant located ...  
 USR3 [*bares in*] ... in the city centre.  
 SYS3 *inform(area=centre [0.8])*

Table 4: Example dialogue where the dialogue manager needs to send incremental updates to the NLG. Incremental surface realisation from semantic trees for this dialogue is shown in Figure 3.

because the authors tested only for *Phrasing* and *Natural*, respectively.

## 5 Incremental Surface Realisation

Recent years have seen increased interest in incremental dialogue processing (Skantze and Schlangen, 2009; Schlangen and Skantze, 2009). The main characteristic of incremental architectures is that instead of waiting for the end of a user turn, they begin to process the input stream as soon as possible, updating their processing hypotheses as more information becomes available. From a dialogue perspective, they can be said to work on partial rather than full dialogue acts.

With respect to surface realisation, incremental NLG systems have predominantly relied on pre-defined templates (Purver and Otsuka, 2003; Skantze and Hjalmarsson, 2010; Dethlefs et al., 2012a), which limits the flexibility and quality of output generation. Buschmeier et al. (2012) have presented a system which systematically takes the user's acoustic understanding problems into account by pausing, repeating or re-phrasing if necessary. Their approach is based on SPUD (Stone et al., 2003), a constraint satisfaction-based NLG architecture and marks important progress towards more flexible incremental surface realisation. However, given the human labour involved in constraint specification, cohesion is often limited to a local context. Especially for long utterances or such that are separated by user turns, this may lead to surface form increments that are not well connected and lack cohesion.

### 5.1 Application to Incremental SR

This section will discuss a proof-of-concept application of our approach to incremental surface realisation. Table 4 shows an example dialogue between a user and system that contains a number of incremental phenomena that require hypothesis updates, system corrections and user barge-ins. Incremental surface realisation for this dialogue is shown in Figure 3, where processing steps are indicated as bold-face numbers and are triggered by partial dialogue acts that are sent from the dialogue manager, such as *inform(area=centre [0.2])*. The numbers in square brackets indicate the system's confidence in the attribute-value pair. Once a dialogue act is observed by the NLG system, a reinforcement learning agent determines the order of attributes and produces a semantic tree, as described in Section 3.1. Since the semantic forms are constructed incrementally, new tree nodes can be attached to and deleted from an existing tree, depending on what kind of update is required.

In the dialogue in Table 4, the user first asks for a *nice restaurant in the centre*. The dialogue manager constructs a first attribute-value slot, *inform(area=centre [0.2], ...)*, and passes it on to NLG.<sup>7</sup> In Figure 3, we can observe the corresponding NLG action, a first tree is created with just a root node and a node representing the area slot (step 1). In a second step, the semantically annotated node gets **expanded** into a surface form that is chosen from a set of candidates (shown in curly brackets). The CRF is responsible for this last step. Since there is no preceding utterance, the best surface form is chosen based on the semantics alone. Active tree nodes, i.e. those currently under generation, are indicated as asterisks in Figure 3. Currently inactive nodes are shown as circles.

Step 3 then further expands the current tree adding a node for the food type and the name of a restaurant that the dialogue manager had passed. We see here that attributes can either be primitive or complex. Primitive attributes contain a single semantic type, such as *area*, whereas complex attributes contain multiple types, such as *food*, *name* and need to be decomposed in a later processing step (see steps 4 and 6). Step 5 again uses the CRF

<sup>7</sup>Note here that the information passed on to the NLG is distinct from the dialogue manager's own actions. In the example, the NLG is asked to generate a recommendation, but the dialogue manager actually decides to clarify the user's preferences due to low confidence. This scenario is an example of generator inputs that may get revised afterwards.

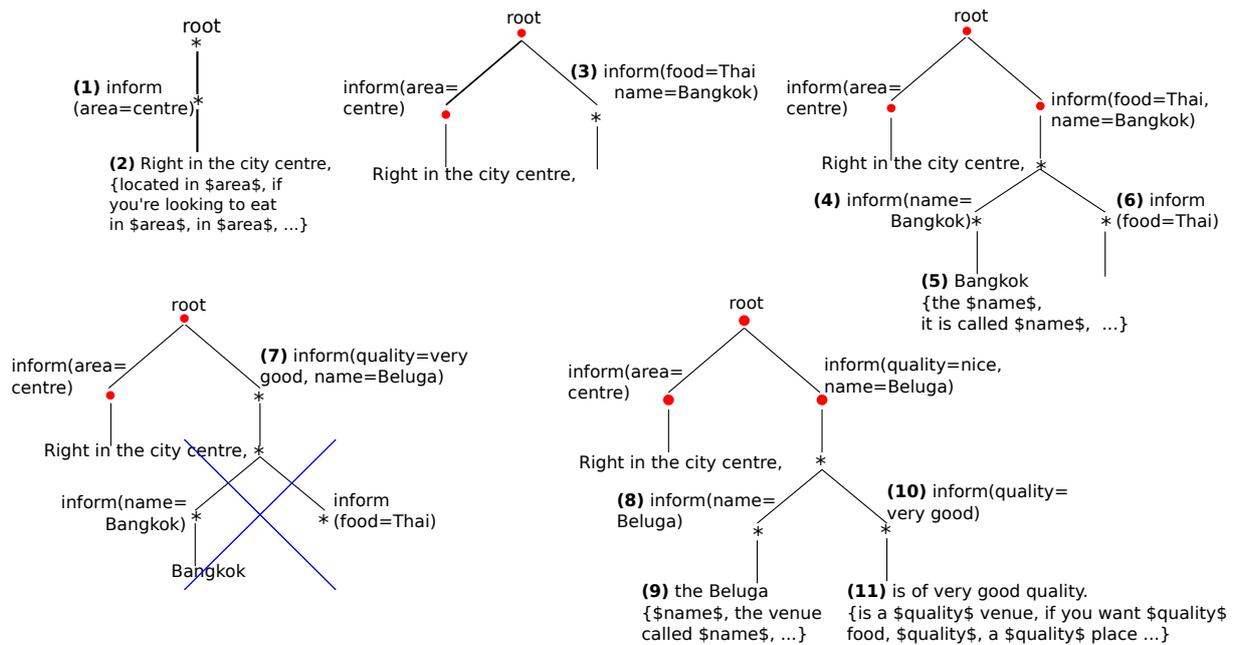


Figure 3: Example of incremental surface realisation, where each generation step is indicated by a number. Active generation nodes are shown as asterisks and deletions are shown as crossed out. Lexical and semantic features are associated with their respective nodes. Syntactic information in the form of parse categories are also taken into account for surface realisation, but have been omitted in this figure.

to obtain the next surface realisation that connects with the previous one (so that a sequence of realisation “labels” appears: *Right in the city centre* and *Bangkok*). It takes the full generation context into account to ensure a globally optimal choice. This is important, because the local context would otherwise be restricted to a partial dialogue act, which can be much smaller than a full dialogue act and thus lead to short, repetitive sentences.

The dialogue continues as the system implicitly confirms the user’s preferred restaurant (SYS1). At this point, we encounter a user barge-in correcting the desired choice. As a consequence, the dialogue manager needs to **update** its initial hypotheses and communicate this to NLG. Here, the last three tree nodes need to be **deleted** from the tree because the information is no longer valid. This update and the deletion is shown in step 7. Afterwards, the dialogue continues and NLG involves mainly expanding the current tree into a full sequence of surface realisations for partial dialogue acts which come together into a full utterance.

This example illustrates three incremental processing steps: expansions, updates and deletions. **Expansions** are the most frequent operation. They add new partial dialogue acts to the semantic tree. They also consult the CRF for the best surface

realisation. Since CRFs are not restricted by the Markov condition, they are less constrained by local context than other models and can take non-local dependencies into account. For our application, the maximal context is 9 semantic attributes (for a surface form that uses all possible 10 attributes). While their extended context awareness can often make CRFs slow to train, they are fast at execution and therefore very applicable to the incremental scenario. For applications involving longer-spanning alternatives, such as texts or paragraphs, the context of the CRF would likely have to be constrained. **Updates** are triggered by the hypothesis updates of the dialogue manager. Whenever a new attribute comes in, it is checked against the generator’s existing knowledge. If it is inconsistent with previous knowledge, an update is triggered and often followed by a **deletion**. Whenever generated output needs to be modified, old expansions and surface forms are deleted first, before new ones can be expanded in their place.

## 5.2 Updates and Processing Speed Results

Since fast responses are crucial in incremental systems, we measured the average time our system took for a surface realisation. The time is 100ms on a MacBook Intel Core 2.6 Duo with 8GB in

RAM. This is slightly better than other incremental systems (Skantze and Schlangen, 2009) and much faster than state-of-the-art non-incremental systems such as SPaRKY (Walker et al., 2007). In addition, we measured the number of necessary generation updates in comparison to a non-incremental setting. Since updates take effect directly on partial dialogue acts, rather than the full generated utterance, we require around 50% less updates as if generating from scratch for every changed input hypothesis. A qualitative analysis of the generated outputs showed that the quality is comparable to the non-incremental case.

## 6 Conclusion and Future Directions

We have presented a novel technique for surface realisation that treats generation as a sequence labelling task by combining a CRF with tree-based semantic representations. An essential property of interactive surface realisers is to keep track of the utterance context including dependencies between linguistic features to generate cohesive utterances. We have argued that CRFs are well suited for this task because they are not restricted by independence assumptions. In a human rating study, we confirmed that judges rated our output as better phrased, more natural and less repetitive than systems that just take local features into account. This also holds for a comparison with state-of-the-art rank and boost or  $n$ -gram approaches. Keeping track of the global context is also important for incremental systems since generator inputs can be incomplete or subject to modification. In a proof-of-concept study, we have argued that our approach is applicable to incremental surface realisation. This was supported by preliminary results on the speed, number of updates and quality during generation. As future work, we plan to test our model in a task-based setting using an end-to-end SDS in an incremental and non-incremental setting. This study will contain additional evaluation categories, such as the understandability or informativeness of system utterances. In addition, we may compare different sequence labelling algorithms for surface realisation (Nguyen and Guo, 2007) or segmented CRFs (Sarawagi and Cohen, 2005) and apply our method to more complex surface realisation domains such as text generation or summarisation. Finally, we would like to explore methods for unsupervised data labelling so as to facilitate portability across domains further.

## Acknowledgements

The research leading to this work was funded by the EC FP7 programme FP7/2011-14 under grant agreement no. 287615 (PARLANCE).

## References

- Ivan Bulyko and Mari Ostendorf. 2002. Efficient integrated response generation from multiple targets using weighted finite state transducers. *Computer Speech and Language*, 16:533–550.
- Hendrik Buschmeier, Timo Baumann, Benjamin Dosch, Stefan Kopp, and David Schlangen. 2012. Incremental Language Generation and Incremental Speech Synthesis. In *Proceedings of the 13th Annual SigDial Meeting on Discourse and Dialogue (SIGdial)*, Seoul, South Korea.
- Nina Dethlefs and Heriberto Cuayáhuitl. 2011a. Combining Hierarchical Reinforcement Learning and Bayesian Networks for Natural Language Generation in Situated Dialogue. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.
- Nina Dethlefs and Heriberto Cuayáhuitl. 2011b. Hierarchical Reinforcement Learning and Hidden Markov Models for Task-Oriented Natural Language Generation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, Portland, Oregon, USA.
- Nina Dethlefs, Helen Hastie, Verena Rieser, and Oliver Lemon. 2012a. Optimising Incremental Dialogue Decisions Using Information Density for Interactive Systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL)*, Jeju, South Korea.
- Nina Dethlefs, Helen Hastie, Verena Rieser, and Oliver Lemon. 2012b. Optimising Incremental Generation for Spoken Dialogue Systems: Reducing the Need for Fillers. In *Proceedings of the International Conference on Natural Language Generation (INLG)*, Chicago, Illinois, USA.
- Kallirroi Georgila, Nikos Fakotakis, and George Kokkinakis. 2002. Stochastic Language Modelling for Recognition and Generation in Dialogue Systems. *TAL (Traitement automatique des langues) Journal*, 43(3):129–154.
- Ioannis Konstas and Mirella Lapata. 2012. Concept-to-text Generation via Discriminative Reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 369–378, Jeju Island, Korea.
- John D. Lafferty, Andrew McCallum, and Fernando C.N. Pereira. 2001. Conditional Random

- Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289.
- Wei Lu, Hwee Tou Ng, and Wee Sun Lee. 2009. Natural Language Generation with Tree Conditional Random Fields. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Singapore.
- François Mairesse, Filip Jurčićek, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. 2010. Phrase-Based Statistical Language Generation Using Graphical Models and Active Learning. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics (ACL)*, Uppsala, Sweden.
- Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, Genoa, Italy.
- Andrew McCallum. 2002. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- Crystal Nakatsu and Michael White. 2006. Learning to Say It Well: Reranking Realizations by Predicted Synthesis Quality. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (COLING-ACL) 2006*, pages 1113–1120, Sydney, Australia.
- Nam Nguyen and Yunsong Guo. 2007. Comparisons of Sequence Labeling Algorithms and Extensions. In *Proceedings of the International Conference on Machine Learning (ICML)*, Corvallis, OR, USA.
- Alice Oh and Alexander Rudnicky. 2000. Stochastic Language Generation for Spoken Dialogue Systems. In *Proceedings of the ANLP/NAACL Workshop on Conversational Systems*, pages 27–32, Seattle, Washington, USA.
- Matthew Purver and Masayuki Otsuka. 2003. Incremental Generation by Incremental Parsing. In *Proceedings of the 6th UK Special-Interesting Group for Computational Linguistics (CLUK) Colloquium*.
- Verena Rieser, Simon Keizer, Xingkun Liu, and Oliver Lemon. 2011. Adaptive Information Presentation for Spoken Dialogue Systems: Evaluation with Human Subjects. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.
- Sunita Sarawagi and William Cohen. 2005. Semi-Markov Conditional Random Fields for Information Extraction. *Advances in Neural Information Processing*.
- David Schlangen and Gabriel Skantze. 2009. A General, Abstract Model of Incremental Dialogue Processing. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, Athens, Greece.
- Gabriel Skantze and Anna Hjalmarsson. 2010. Towards Incremental Speech Generation in Dialogue Systems. In *Proceedings of the 11th Annual SigDial Meeting on Discourse and Dialogue*, Tokyo, Japan.
- Gabriel Skantze and David Schlangen. 2009. Incremental Dialogue Processing in a Micro-Domain. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, Athens, Greece.
- Amanda Stent, Marilyn Walker, Steve Whittaker, and Preetam Maloor. 2002. User-tailored Generation for Spoken Dialogue: An Experiment. In *Proceedings of the International Conference on Spoken Language Processing*.
- Matthew Stone, Christine Doran, Bonnie Webber, Tonia Bleam, and Martha Palmer. 2003. Microplanning with Communicative Intentions: The SPUD System. *Computational Intelligence*, 19:311–381.
- Charles Sutton and Andrew McCallum. 2006. Introduction to Conditional Random Fields for Relational Learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.
- Sebastian Varges. 2006. Overgeneration and Ranking for Spoken Dialogue Systems. In *Proceedings of the Fourth International Natural Language Generation Conference (INLG)*, Sydney, Australia.
- Marilyn Walker, Amanda Stent, François Mairesse, and Rashmi Prasad. 2007. Individual and Domain Adaptation in Sentence Planning for Dialogue. *Journal of Artificial Intelligence Research*, 30(1):413–456.

# Two-Neighbor Orientation Model with Cross-Boundary Global Contexts

Hendra Setiawan, Bowen Zhou, Bing Xiang and Libin Shen

IBM T.J.Watson Research Center

1101 Kitchawan Road

Yorktown Heights, NY 10598, USA

{hendras, zhou, bxiang, lshen}@us.ibm.com

## Abstract

Long distance reordering remains one of the greatest challenges in statistical machine translation research as the key contextual information may well be beyond the confine of translation units. In this paper, we propose Two-Neighbor Orientation (TNO) model that jointly models the orientation decisions between anchors and two neighboring multi-unit chunks which may cross phrase or rule boundaries. We explicitly model the longest span of such chunks, referred to as Maximal Orientation Span, to serve as a global parameter that constrains underlying local decisions. We integrate our proposed model into a state-of-the-art string-to-dependency translation system and demonstrate the efficacy of our proposal in a large-scale Chinese-to-English translation task. On NIST MT08 set, our most advanced model brings around +2.0 BLEU and -1.0 TER improvement.

## 1 Introduction

Long distance reordering remains one of the greatest challenges in Statistical Machine Translation (SMT) research. The challenge stems from the fact that an accurate reordering hinges upon the model's ability to make many local and global reordering decisions accurately. Often, such reordering decisions require contexts that span across multiple translation units.<sup>1</sup> Unfortunately, previous approaches fall short in capturing such cross-unit contextual information that could be

<sup>1</sup>We define translation units as phrases in phrase-based SMT, and as translation rules in syntax-based SMT.

critical in reordering. Specifically, the popular distortion or lexicalized reordering models in phrase-based SMT focus only on making good local prediction (i.e. predicting the orientation of immediate neighboring translation units), while translation rules in syntax-based SMT come with a strong context-free assumption, which model only the reordering within the confine of the rules. In this paper, we argue that reordering modeling would greatly benefit from richer cross-boundary contextual information

We introduce a reordering model that incorporates such contextual information, named the Two-Neighbor Orientation (TNO) model. We first identify *anchors* as regions in the source sentences around which ambiguous reordering patterns frequently occur and *chunks* as regions that are consistent with word alignment which may span multiple translation units at decoding time. Most notably, anchors and chunks in our model may not necessarily respect the boundaries of translation units. Then, we jointly model the orientations of chunks that immediately precede and follow the anchors (hence, the name “two-neighbor”) along with the maximal span of these chunks, to which we refer as Maximal Orientation Span (MOS).

As we will elaborate further in next sections, our models provide a stronger mechanism to make more accurate global reordering decisions for the following reasons. First of all, we consider the orientation decisions on both sides of the anchors simultaneously, in contrast to existing works that only consider one-sided decisions. In this way, we hope to upgrade the unigram formulation of existing reordering models to a higher order formulation. Second of all, we capture the reordering of chunks that may cross translation units and may be composed of multiple units, in contrast to ex-

isting works that focus on the reordering between individual translation units. In effect, MOS acts as a global reordering parameter that guides or constrains the underlying local reordering decisions.

To show the effectiveness of our model, we integrate our TNO model into a state-of-the-art syntax-based SMT system, which uses synchronous context-free grammar (SCFG) rules to jointly model reordering and lexical translation. The introduction of nonterminals in the SCFG rules provides some degree of generalization. However as mentioned earlier, the context-free assumption ingrained in the syntax-based formalism often limits the model’s ability to influence global reordering decision that involves cross-boundary contexts. In integrating TNO, we hope to strengthen syntax-based system’s ability to make more accurate global reordering decisions.

Our other contribution in this paper is a practical method for integrating the TNO model into syntax-based translations. The integration is non-trivial since the decoding of syntax-based SMT proceeds in a bottom-up fashion, while our model is more natural for top-down parsing, thus the model’s full context sometimes is often available only at the latest stage of decoding. We implement an efficient shift-reduce algorithm that facilitates the accumulation of partial context in a bottom-up fashion, allowing our model to influence the translation process even in the absence of full context.

We show the efficacy of our proposal in a large-scale Chinese-to-English translation task where the introduction of our TNO model provides a significant gain over a state-of-the-art string-to-dependency SMT system (Shen et al., 2008) that we enhance with additional state-of-the-art features. Even though the experimental results carried out in this paper employ SCFG-based SMT systems, we would like to point out that our models is applicable to other systems including phrase-based SMT systems.

The rest of the paper is organized as follows. In Section 2, we introduce the formulation of our TNO model. In Section 3, we introduce and motivate the concept of Maximal Orientation Span. In Section 4, we introduce four variants of the TNO model with different model complexities. In Section 5, we describe the training procedure to estimate the parameters of our models. In Section 6, we describe our shift-reduce algorithm which inte-

grates our proposed TNO model into syntax-based SMT. In Section 7, we describe our experiments and present our results. We wrap up with related work in Section 8 and conclusion in Section 9.

## 2 Two-Neighbor Orientation Model

Given an aligned sentence pair  $\Theta = (F, E, \sim)$ , let  $\Delta(\Theta)$  be all possible chunks that can be extracted from  $\Theta$  according to:<sup>2</sup>

$$\{(f_{j_1}^{j_2}/e_{i_1}^{i_2}) : \forall j_1 \leq j_2, \exists i : (j, i) \in \sim, i_i \leq i \leq i_2 \wedge \forall i_1 \leq i \leq i_2, \exists j : (j, i) \in \sim, j_i \leq j \leq j_2\}$$

Our Two-Neighbor Orientation model (TNO) designates  $\mathcal{A} \subset \Delta(\Theta)$  as anchors and *jointly* models the orientation of chunks that appear *immediately* to the left and to the right of the anchors as well as the identities of these chunks. We define anchors as chunks, around which ambiguous reordering patterns frequently occur. Anchors can be learnt automatically from the training data or identified from the linguistic analysis of the source sentence. In our experiments, we use a simple heuristics based on part-of-speech tags which will be described in Section 7.

More concretely, given  $\mathcal{A} \subset \Delta(\Theta)$ , let  $a = (f_{j_1}^{j_2}/e_{i_1}^{i_2}) \in \mathcal{A}$  be a particular anchor. Then, let  $\mathcal{C}_L(a) \subset \Delta(\Theta)$  be  $a$ ’s left neighbors and let  $\mathcal{C}_R(a) \subset \Delta(\Theta)$  be  $a$ ’s right neighbors, iff:

$$\forall \mathcal{C}_L = (f_{j_3}^{j_4}/e_{i_3}^{i_4}) \in \mathcal{C}_L(a) : j_4 + 1 = j_1 \quad (1)$$

$$\forall \mathcal{C}_R = (f_{j_5}^{j_6}/e_{i_5}^{i_6}) \in \mathcal{C}_R(a) : j_2 + 1 = j_5 \quad (2)$$

Given  $\mathcal{C}_L(a)$  and  $\mathcal{C}_R(a)$ , let  $C_L = (f_{j_3}^{j_4}/e_{i_3}^{i_4})$  and  $C_R = (f_{j_5}^{j_6}/e_{i_5}^{i_6})$  be a particular pair of left and right neighbors of  $a = (f_{j_1}^{j_2}/e_{i_1}^{i_2})$ . Then, the orientation of  $C_L$  and  $C_R$  are  $O_L(C_L, a)$  and  $O_R(C_R, a)$  respectively and each may take one of the following four orientation values (similar to (Nagata et al., 2006)):

- **Monotone Adjacent (MA)**, if  $(i_4 + 1) = i_1$  for  $O_L$  and if  $(i_2 + 1) = i_5$  for  $O_R$
- **Reverse Adjacent (RA)**, if  $(i_2 + 1) = i_3$  for  $O_L$  and if  $(i_6 + 1) = i_1$  for  $O_R$
- **Monotone Gap (MG)**, if  $(i_4 + 1) < i_1$  for  $O_L$  and if  $(i_2 + 1) < i_5$  for  $O_R$

<sup>2</sup>We represent a chunk as a source and target phrase pair  $(f_{j_1}^{j_2}/e_{i_1}^{i_2})$  where the subscript and the superscript indicate the starting and the ending indices as such  $f_{j_1}^{j_2}$  denotes a source phrase that spans from  $j_1$  to  $j_2$ .

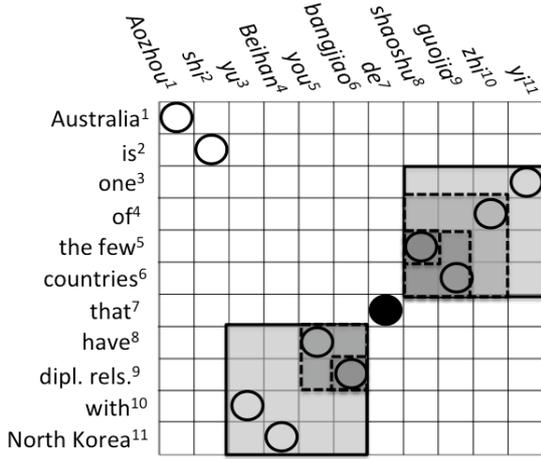


Figure 1: An aligned Chinese-English sentence pair. Circles represent alignment points. Black circle represents the anchor; boxes represent the anchor’s neighbors.

- **Reverse Gap (RG)**, if  $(i_2 + 1) < i_3$  for  $O_L$  and if  $(i_6 + 1) < i_1$  for  $O_R$ . (1)

The first clause (monotone, reverse) indicates whether the target order of the chunks follows the source order; the second (adjacent, gap) indicates whether the chunks are adjacent or separated by an intervening phrase when projected.

To be more concrete, let us consider an aligned sentence pair in Fig. 1, which is adapted from (Chiang, 2005). Suppose there is only one anchor, i.e.  $a = (f_7^7/e_7^7)$  which corresponds to the word *de(that)*. By applying Eqs. 1 and 2, we can infer that  $a$  has three left neighbors and four right neighbors, i.e.  $C_L(a) = (f_6^6/e_9^9), (f_5^6/e_8^9), (f_3^6/e_8^{11})$  and  $C_R(a) = (f_8^8/e_5^5), (f_8^9/e_5^6), (f_8^{10}/e_4^6), (f_8^{11}/e_3^6)$  respectively. Then, by applying Eq. 1, we can compute the orientation values of each of these neighbors, which are  $O_L(C_L(a), a) = RG, RA, RA$  and  $O_R(C_R(a), a) = RG, RA, RA, RA$ . As shown, most of the neighbors have Reverse Adjacent (RA) orientation except for the smallest left and right neighbors (i.e.  $(f_6^6/e_9^9)$  and  $(f_8^8/e_5^5)$ ) which have Reverse Gap (RG) orientation.

Given the anchors together with its neighboring chunks and their orientations, the Two-Neighbor Orientation model takes the following form:

$$\prod_{a \in \mathcal{A}} \sum_{\substack{C_L \in \mathcal{C}_L(a), \\ C_R \in \mathcal{C}_R(a)}} P_{TNO}(C_L, O_L, C_R, O_R | a; \Theta) \quad (2)$$

For conciseness, references that are clear from context, such the reference to  $C_L$  and  $a$  in  $O_L(C_L, a)$ , are dropped.

### 3 Maximal Orientation Span

As shown in Eq. 2, the TNO model has to enumerate all possible pairing of  $C_L \in \mathcal{C}_L(a)$  and  $C_R \in \mathcal{C}_R(a)$ . To make the TNO model more tractable, we simplify the TNO model to consider only the largest left and right neighbors, referred to as the Maximal Orientation Span/MOS ( $M$ ).

More formally, given  $a = (f_{j_1}^{j_2}/e_{i_1}^{i_2})$ , the left and the right MOS of  $a$  are:

$$M_L(a) = \arg \max_{(f_{j_3}^{j_4}/e_{i_3}^{i_4}) \in \mathcal{C}_L(a)} (j_4 - j_3)$$

$$M_R(a) = \arg \max_{(f_{j_5}^{j_6}/e_{i_5}^{i_6}) \in \mathcal{C}_R(a)} (j_6 - j_5)$$

Coming back to our example, the left and right MOS of the anchor are  $M_L(a) = (f_3^6/e_8^{11})$  and  $M_R(a) = (f_8^{11}/e_3^6)$ . In Fig. 1, they are denoted as the largest boxes delineated by solid lines.

As such, we reformulate Eq. 2 into:

$$\prod_{a \in \mathcal{A}} \sum_{\substack{C_L \in \mathcal{C}_L(a), \\ C_R \in \mathcal{C}_R(a)}} P_{TNO}(M_L, O_L, M_R, O_R | a; \Theta) \cdot \delta_{C_L == M_L \wedge C_R == M_R} \quad (3)$$

where  $\delta$  returns 1 if  $(C_L == M_L \wedge C_R == M_R)$ , otherwise 0.

Beyond simplifying the computation, the key benefit of modeling MOS is that it serves as a global parameter that can guide or constrain underlying local reorderings. As a case in point, let us consider a cheating exercise where we have to translate the Chinese sentence in Fig. 1 with the following set of hierarchical phrases<sup>3</sup>:

$$\begin{aligned} X_a &\rightarrow \langle \text{Aozhou}^1 \text{shi}^2 X_1, \text{Australia}^1 \text{is}^2 X_1 \rangle \\ X_b &\rightarrow \langle \text{yu}^3 \text{Beihan}^4 X_1, X_1 \text{with}^3 \text{North}^4 \text{Korea} \rangle \\ X_c &\rightarrow \langle \text{you}^5 \text{bangjiao}^6, \text{have}^5 \text{dipl.}^6 \text{rels.} \rangle \\ X_d &\rightarrow \langle X_1 \text{de}^7 \text{shaoshu}^8 \text{guojia}^9 \text{zhi}^{10} \text{yi}^{11}, \\ &\quad \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \text{that}^7 X_1 \rangle \end{aligned}$$

This set of hierarchical phrases represents a translation model that has resolved all local ambiguities (i.e. local reordering and lexical mappings) except for the spans of the hierarchical phrases. With this example, we want to show that accurate local decisions (rather obviously) don’t always lead to accurate global reordering and to demonstrate that explicit MOS modeling can play a crucial role to address this issue. To do so, we will again focus on the same anchor *de* (that).

<sup>3</sup>We use hierarchical phrase-based translation system as a case in point, but the merit is generalizable to other systems.

$$\begin{aligned}
&\stackrel{d}{\Rightarrow} \langle X_1 \mathbf{de}^7 \mathbf{shaoshu}^8 \mathbf{guojia}^9 \mathbf{zhi}^{10} \mathbf{yi}^{11} \rangle, \langle \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \mathbf{that}^7 X_1 \rangle \\
&\stackrel{a}{\Rightarrow} \langle \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 X_1 \rangle \mathbf{de}^7 \mathbf{shaoshu}^8 \mathbf{guojia}^9 \mathbf{zhi}^{10} \mathbf{yi}^{11} \rangle, \\
&\quad \langle \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \mathbf{that}^7 \langle \text{Australia}^1 \text{is}^2 X_1 \rangle \rangle \\
&\stackrel{b}{\Rightarrow} \langle \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 \langle \mathbf{yu}^3 \mathbf{Beihan}^4 X_1 \rangle \rangle \mathbf{de}^7 \mathbf{shaoshu}^8 \mathbf{guojia}^9 \mathbf{zhi}^{10} \mathbf{yi}^{11} \rangle, \\
&\quad \langle \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \mathbf{that}^7 \langle \text{Australia}^1 \text{is}^2 \langle X_1 \text{with}^3 \text{North}^4 \text{Korea} \rangle \rangle \rangle \\
&\stackrel{c}{\Rightarrow} \boxed{\langle d \rangle} \langle \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 \langle \mathbf{yu}^3 \mathbf{Beihan}^4 \langle \mathbf{c} \mathbf{you}^5 \mathbf{bangjiao}^6 \rangle \mathbf{c} \rangle \mathbf{de}^7 \mathbf{shaoshu}^8 \mathbf{guojia}^9 \mathbf{zhi}^{10} \mathbf{yi}^{11} \rangle \boxed{\langle d \rangle} \rangle, \\
&\quad \langle \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \mathbf{that}^7 \langle \text{Australia}^1 \text{is}^2 \langle \langle \text{have}^5 \text{dipl.}^6 \text{reels.} \rangle \text{with}^3 \text{North}^4 \text{Korea} \rangle \rangle \rangle
\end{aligned}$$

Table 1: Derivation of  $X_d \prec X_a \prec X_b \prec X_c$  that leads to an incorrect translation.

$$\begin{aligned}
&\stackrel{a}{\Rightarrow} \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 X_1 \rangle, \langle \text{Australia}^1 \text{is}^2 X_1 \rangle \\
&\stackrel{b}{\Rightarrow} \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 \langle \mathbf{yu}^3 \mathbf{Beihan}^4 X_1 \rangle \rangle, \langle \text{Australia}^1 \text{is}^2 \langle X_1 \text{with}^3 \text{North}^4 \text{Korea} \rangle \rangle \\
&\stackrel{d}{\Rightarrow} \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 \langle \mathbf{yu}^3 \mathbf{Beihan}^4 \langle X_1 \mathbf{de}^7 \mathbf{shaoshu}^8 \mathbf{guojia}^9 \mathbf{zhi}^{10} \mathbf{yi}^{11} \rangle \rangle \rangle, \\
&\quad \langle \text{Australia}^1 \text{is}^2 \langle \langle \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \mathbf{that}^7 X_1 \rangle \text{with}^3 \text{North}^4 \text{Korea} \rangle \rangle \rangle \\
&\stackrel{c}{\Rightarrow} \langle \mathbf{Aozhou}^1 \mathbf{shi}^2 \langle \mathbf{yu}^3 \mathbf{Beihan}^4 \langle \boxed{\langle d \rangle} \langle \mathbf{c} \mathbf{you}^5 \mathbf{bangjiao}^6 \rangle \mathbf{c} \mathbf{de}^7 \mathbf{shaoshu}^8 \mathbf{guojia}^9 \mathbf{zhi}^{10} \mathbf{yi}^{11} \rangle \boxed{\langle d \rangle} \rangle \rangle, \\
&\quad \text{Australia}^1 \text{is}^2 \langle \langle \text{one}^{11} \text{of}^{10} \text{the few}^8 \text{countries}^9 \mathbf{that}^7 \langle \text{have}^5 \text{dipl.}^6 \rangle \text{with}^3 \text{North}^4 \text{Korea} \rangle \rangle \rangle
\end{aligned}$$

Table 2: Derivation of  $X_a \prec X_b \prec X_d \prec X_c$  that leads to the correct translation.

As the rule’s identifier, we attach an alphabet letter to each rule’s left hand side, as such the anchor *de* (that) appears in rule  $X_d$ . We also attach the word indices as the superscript of the source words and project the indices to the target words aligned, as such “*have*<sup>5</sup>” suggests that the word “have” is aligned to the 5-th source word, i.e. *you*. Note that to facilitate the projection, the rules must come with internal word alignment in practice. Now the indices on the target words in the rules are different from those in Fig. 1. We will also extensively use indices in this sense in the subsequent section about decoding. In such a sense,  $M_L(a) = (f_3^6/e_3^6)$  and  $M_R(a) = (f_8^{11}/e_8^{11})$ .

Given the rule set, there are three possible derivations, i.e.  $X_d \prec X_a \prec X_b \prec X_c$ ,  $X_a \prec X_b \prec X_d \prec X_c$ , and  $X_a \prec X_d \prec X_b \prec X_c$ , where  $\prec$  indicates that the first operand dominates the second operand in the derivation tree. The application of the rules would show that the first derivation will produce an incorrect reordering while the last two will produce the correct ones. Here, we would like to point out that even in this simple example where all local decisions are made accurate, this ambiguity occurs and it would occur even more so in the real translation task where local decisions may be highly inaccurate.

Next, we will show that the MOS-related information can help to resolve this ambiguity, by focusing more closely on the first and the second derivations, which are detailed in Tables 1 and 2.

Particularly, we want to show that the MOS generated by the incorrect derivation does not match the MOS learnt from Fig. 1. As shown, at the end of the derivation, we have all the information needed to compute the MOS (i.e.  $\Theta$ ) which is equivalent to that available at training time, i.e. the source sentence, the complete translation and the word alignment. Running the same MOS extraction procedure on both derivations would produce the right MOS that agrees with the right MOS previously learnt from Fig. 1, i.e.  $(f_8^{11}/e_8^{11})$ . However, that’s not the case for left MOS, which we underline in Tables 1 and 2. As shown, the incorrect derivation produces a left MOS that spans six words, i.e.  $(f_1^6/e_1^6)$ , while the correct derivation produces a left MOS that spans four words, i.e.  $(f_3^6/e_3^6)$ . Clearly, the MOS of the incorrect derivation doesn’t agree with the MOS we learnt from Fig. 1, unlike the MOS of the correct translation. This suggests that explicit MOS modeling would provide a mechanism for resolving crucial global reordering ambiguities that are beyond the ability of local models.

Additionally, this illustration also shows a case where MOS acts as a cross-boundary context which effectively relaxes the context-free assumption of hierarchical phrase-based formalism. In Tables 1 and 2’s full derivations, we indicate rule boundaries explicitly by indexing the angle brackets, e.g.  $\langle_a$  indicates the beginning of rule  $X_a$  in the derivation. As the anchor appears in  $X_d$ , we

highlight its boundaries in box frames. *de* (that)’s MOS respects rule boundaries if and only if all the words come entirely from  $X_d$ ’s antecedent or  $\langle_d$  and  $\rangle_d$  appears outside of MOS; otherwise it crosses the rule boundaries. As clearly shown in Table 2, the left MOS of the correct derivation (underlined) crosses the rule boundary (of  $X_d$ ) since  $\langle_d$  appears within the MOS.

Going back to the formulation, focusing on modeling MOS would simplify the formulation of TNO model from Eq. 2 into:

$$\prod_{a \in \mathcal{A}} P_{TNO}(M_L, O_L, M_R, O_R | a; \Theta) \quad (4)$$

which doesn’t require enumerating of all possible pairs of  $\mathcal{C}_L$  and  $\mathcal{C}_R$ .

#### 4 Model Decomposition and Variants

To make the model more tractable, we decompose  $P_{TNO}$  in Eq. 4 into the following four factors:  $P(M_R | a) \times P(O_R | M_R, a) \times P(M_L | O_R, M_R, a) \times P(O_L | M_L, O_R, M_R, a)$ . Subsequently, we will refer to them as  $P_{M_R}$ ,  $P_{O_R}$ ,  $P_{M_L}$  and  $P_{O_L}$  respectively. Each of these factors will act as an additional feature in the log-linear framework of our SMT system. The above decomposition follows a generative story that starts from generating the right neighbor first. There are other equally credible alternatives, but based on empirical results, we settle with the above.

Next, we present four different variants of the model (not to be confused with the four factors above). Each variant has a different probabilistic conditioning of the factors. We start by making strong independence assumptions in Model 1 and then relax them as we progress to Model 4. The description of the models is as follow:

- **Model 1.** We assume  $P_{M_L}$  and  $P_{M_R}$  to be equal to 1 and  $P_{O_R} \approx P(O_R | a; \Theta)$  to be independent of  $M_R$  and  $P_{O_L} \approx P(O_L | a; \Theta)$  to be independent of  $M_L, M_R$  and  $O_R$ .
- **Model 2.** On top of Model 1, we make  $P_{O_L}$  dependent on  $P_{O_R}$ , thus  $P_{O_L} \approx P(O_L | O_R, a; \Theta)$ .
- **Model 3.** On top of Model 2, we make  $P_{O_R}$  dependent on  $M_R$  and  $P_{O_L}$  on  $M_R$  and  $M_L$ , thus  $P_{O_R} \approx P(O_R | M_R, a; \Theta)$  and  $P_{O_L} \approx P(O_L | M_L, O_R, M_R; a, \Theta)$ .
- **Model 4.** On top of Model 3, we model  $P_{M_R}$  and  $P_{M_L}$  as multinomial distributions estimated from training data.

Model 1 represents a model that focuses on making accurate one-sided decisions, independent of the decision on the other side. Model 2 is designed to address the deficiency of Model 1 since Model 1 may assign non-zero probability to improbable assignment of orientation values, e.g. Monotone Adjacent for the left neighbor and Reverse Adjacent for the right neighbor. Model 2 does so by conditioning  $P_{O_L}$  on  $O_R$ . In Model 3, we start incorporating MOS-related information in predicting  $O_L$  and  $O_R$ . In Model 4, we explicitly model the MOS of each anchor.

#### 5 Training

The TNO model training consists of two different training regimes: 1) discriminative for training  $P_{O_L}, P_{O_R}$ ; and 2) generative for training  $P_{M_L}, P_{M_R}$ . Before describing the specifics, we start by describing the procedure to extract anchors and their corresponding MOS from training data, from which we collect statistics and extract features to train the model.

For each aligned sentence pair  $(F, E, \sim)$  in the training data, the training starts with the identification of the regions in the source sentences as anchors ( $\mathcal{A}$ ). For our Chinese-English experiments, we use a simple heuristic that equates as anchors, single-word chunks whose corresponding word class belongs to closed-word classes, bearing a close resemblance to (Setiawan et al., 2007). In total, we consider 21 part-of-speech tags; some of which are as follow: VC (copula), DEG, DEG, DER, DEV (*de*-related), PU (punctuation), AD (adjectives) and P (prepositions).

Next we generate all possible chunks  $\Delta(\Theta)$  as previously described in Sec. 3. We then define a function  $MinC(\Delta, j_1, j_2)$  which returns the shortest chunk that can span from  $j_1$  to  $j_2$ . If  $(f_{j_1}^{j_2} / e_{i_1}^{i_2}) \in \Delta$ , then  $MinC$  returns  $(f_{j_1}^{j_2} / e_{i_1}^{i_2})$ .

The algorithm to extract MOS takes  $\Delta$  and an anchor  $a = (f_{j_1}^{j_2} / e_{i_1}^{i_2})$  as input; and outputs the chunk that qualifies as MOS or none. Alg. 1 provides the algorithm to extract the right MOS; the algorithm to extract the left MOS is identical to Alg. 1, except that it scans for chunks to the left of the anchor. In Alg. 1, there are two intermediate parameters  $si$  and  $ei$  which represent the active search range and should initially be set to  $j_2 + 1$  and  $|F|$  respectively. Once we obtain  $a, M_L(a)$  and  $M_R(a)$ , we compute  $O_L(M_L(a), a)$  and  $O_R(M_R(a), a)$  and are ready for training.

To estimate  $P_{O_L}$  and  $P_{O_R}$ , we train discriminative classifiers that predict the orientation values and use the normalized posteriors at decoding time as additional feature scores in SMT’s log linear framework. We train the classifiers on a rich set of binary features ranging from lexical to part-of-speech (POS) and to syntactic features.

---

**Algorithm 1:** Function  $M_REx$

---

```

input :  $a = (f_{j_1}^{j_2}/e_{i_1}^{i_2}), si, ei$ ; int;  $\Delta$ : chunks
output:  $(f_{j_3}^{j_4}/e_{i_3}^{i_4})$  : chunk or  $\emptyset$ 
 $(f_{j_3}^{j_4}/e_{i_3}^{i_4}) = MinC(\Delta, j_2 + 1, ei)$ 
if  $(j_3 == j_2 + 1 \wedge j_4 == ei)$  then
  |  $\rightarrow f_{j_3}^{j_4}/e_{i_3}^{i_4}$ 
else
  | if  $(j_2 + 1 == ei)$  then
  | |  $\rightarrow \emptyset$ 
  | else
  | | if  $(ei - 2 \leq si)$  then
  | | |  $\rightarrow M_REx(a, si, ei - 1, \Delta)$ 
  | | else
  | | |  $m = \lceil (si + ei) / 2 \rceil$ 
  | | |  $(f_{j_3}^{j_4}/e_{i_4}^{i_4}) = MinC(\Delta, j_2 + 1, m)$ 
  | | | if  $(j_3 == j_2 + 1)$  then
  | | | |  $c = M_REx(a, m, ei - 1, \Delta)$ 
  | | | | if  $(c == \emptyset)$  then
  | | | | |  $\rightarrow f_{j_3}^{j_4}/e_{i_3}^{i_4}$ 
  | | | | else
  | | | | |  $\rightarrow c$ 
  | | | | end
  | | | else
  | | | |  $\rightarrow M_REx(a, si, m - 1, \Delta)$ 
  | | | end
  | | end
  | end
end

```

---

Suppose  $a = (f_{j_1}^{j_2}/e_{i_1}^{i_2})$ ,  $M_L(a) = (f_{j_3}^{j_4}/e_{i_3}^{i_4})$  and  $M_R(a) = (f_{j_5}^{j_6}/e_{i_5}^{i_6})$ , then based on the context’s location, the elementary features employed in our classifiers can be categorized into:

1. *anchor-related*: `slex` (the actual word of  $f_{j_1}^{j_2}$ ), `spos` (part-of-speech (POS) tag of `slex`), `sparent` (`spos`’s parent in the parse tree), `tlex` ( $e_{i_1}^{i_2}$ ’s actual target word)..
2. *surrounding*: `lslex` (the previous word /  $f_{j_1-1}^{j_1-1}$ ), `rslex` (the next word /  $f_{j_2+1}^{j_2+1}$ ), `lspos` (`lslex`’s POS tag), `rspos` (`rslex`’s POS tag), `lsparent` (`lslex`’s parent), `rsparent`

(`rslex`’s parent).

3. *non-local*: `lanchorslex` (the previous anchor’s word), `ranchorslex` (the next anchor’s word), `lanchorspos` (`lanchorslex`’s POS tag), `ranchorspos` (`ranchorslex`’s POS tag).
4. *MOS-related*: `mosl_int_slex` (the actual word of  $f_{j_3}^{j_3}$ ), `mosl_ext_slex` (the actual word of  $f_{j_3}^{j_3}$ ), `mosl_int_spos` (`mosl_int_slex`’s POS tag), `mosl_ext_spos` (`mosl_ext_spos`’s POS tag), `mosr_int_slex` (the actual word of  $f_{j_3}^{j_3}$ ), `mosr_ext_slex` (the actual word of  $f_{j_3}^{j_3}$ ), `mosr_int_spos` (`mosr_int_slex`’s POS tag), `mosr_ext_spos` (`mosr_ext_spos`’s POS tag).

For Model 1, we train one classifier each for  $P_{O_R}$  and  $P_{O_L}$ . For Model 2-4, we train four classifiers for  $P_{O_L}$  for each value of  $O_R$ . We use only the MOS features for Model 3 and 4. Additionally, we augment the feature set with compound features, e.g. conjunction of the lexical of the anchor and the lexical of the left and the right anchors. Although they increase the number of features significantly, we found that these compound features are empirically beneficial.

We come up with  $> 50$  types of features, which consist of a combination of elementary and compound features. In total, we generate hundreds of millions of such features from the training data. To keep the number features to a manageable size, we employ the L1-regularization in training to enforce sparse solutions, using the off-the-shelf LIBLINEAR toolkit (Fan et al., 2008). After training, the number of features in our classifiers decreases to below 5 million features for each classifier.

We train  $P_{M_L}$  and  $P_{M_R}$  via the relative frequency principle. To avoid the sparsity issue, we represent  $M_L$  as (`mosl_int_spos,mosl_ext_spos`) and  $M_R$  as (`mosr_int_spos,mosr_ext_spos`). We condition  $P_{M_L}$  and  $P_{M_R}$  only on `spos` and the orientation, estimating them as follow:

$$P(M_L|spos, O_L) = \frac{N(M_L, spos, O_L)}{N(spos, O_L)}$$

$$P(M_R|spos, O_R) = \frac{N(M_R, spos, O_R)}{N(spos, O_R)}$$

where  $N$  returns the count of the events in the training data.

		Target string (w/ source index)	Symbol(s) read	Op.	Stack(s)
(1)	$X_c$	have <sup>5</sup> dipl. <sup>6</sup> rels.	[5][6]	S,S,R	$X_c$ : [5-6]
(2)	$X_d$	one <sup>11</sup> of <sup>10</sup> few <sup>8</sup> countries <sup>9</sup> <b>that</b> <sup>7</sup> $X_c$	[11][10]	S,S,R	[10-11]
(3)			[8][9]	S,S,R,R	[8-11]
(4)			[7]	S	[8-11][7]
(5)			$X_c$ : [5,6]	S	$X_d$ : [8-11][7][5,6]
(6)	$X_b$	$X_d$ with <sup>3</sup> North <sup>4</sup> Korea	$X_d$ : [8-11][7][5,6]	S	[8-11][7][5,6]
(7)			[3][4]	S,S,R,R	$X_b$ : [8-11][7][3-6]
(8)	$X_a$	Australia <sup>1</sup> is <sup>2</sup> $X_b$	[1][2]	S,S,R	[1-2]
(9)			$X_b$ : [8-11][7][3,6]	S,A	$X_a$ : [1-2][8-11][7][3,6]

Table 3: The application of the shift-reduce parsing algorithm, which corresponds to Table 2’s derivation.

## 6 Decoding

Integrating the TNO Model into syntax-based SMT systems is non-trivial, especially with the MOS modeling. The method described in Sec. 3 assumes  $\Theta = (F, E, \sim)$ , thus it is only applicable at training or at the last stage of decoding. Since many reordering decisions may have been made at the earlier stages, the late application of TNO model would limit the utility of the model. In this section, we describe an algorithm that facilitates the incremental construction of MOS and the computation of TNO model on partial derivations.

The algorithm bears a close resemblance to the shift-reduce algorithm where a stack is used to accumulate (partial) information about  $a$ ,  $M_L$  and  $M_R$  for each  $a \in \mathcal{A}$  in the derivation. This algorithm takes an input stream and applies either the *shift* or the *reduce* operations starting from the beginning until the end of the stream. The *shift* operation advances the input stream by one symbol and push the symbol into the stack; while the *reduce* operation applies some reduction rule to the topmost elements of the stack. The algorithm terminates at the end of the input stream where the resulting stack will be propagated to the parent for the later stage of decoding. In our case, the input stream is the target string of the rule and the symbol is the corresponding source index of the elements of the target string. The reduction rule looks at two indices and merge them if they are adjacent (i.e. has no intervening phrase). We forbid the application of the reduction rule to anchors. Table 3 shows the execution trace of the algorithm for the derivation described in Table 2.

As shown, the algorithm starts with an empty stack. It then projects the source index to the corresponding target word and then enumerates the

target string in a left to right fashion. If it finds a target word with a source index, it applies the shift operation, pushing the index to the stack. Unless the symbol corresponds to an anchor, it tries to apply the reduce operation. Line (4) indicates the special treatment to the anchor. If the symbol read is a nonterminal, then we push the entire stack that corresponds to that nonterminal. For example, when the algorithm reads  $X_d$  at line (6), it pushes the entire stack from line (5).

This algorithm facilitates the incremental construction of MOS which may cross rule boundaries. For example, at the end of the application of  $X_d$  at line (5), the current left MOS is [5-6]. However, the algorithm grows it to [3-6] after the application of rule  $X_b$  at line (7). Furthermore, it allows us to compute the models from partial hypothesis. For example, at line (5), we can compute  $P_{O_L}$  by considering [5,6] as  $M_L$  to be updated with [3,6] in line (7). This way, we expect our TNO model would play a bigger role at decoding time.

Specific to SCFG-based translation, the values of  $O_L$  and  $O_R$  are identical in the partial or in the full derivations. For example, the orientation values of *de* (that)’s left neighbor is always *RA*. This statement holds, even though at the end of Section 2, we stated that *de* (that)’s left neighbor may have other orientation values, i.e. *RG* for  $C_L(a) = (f_6^6/e_9^9)$ . The formal proof is omitted, but the intuition comes from the fact that the derivations for SCFG-based translation are subset of  $\Delta(\Theta)$  and that  $(f_6^6/e_9^9)$  will never become  $M_L$  for  $MinC(C_L(a), a)$  respectively (chunk that spans  $a$  and  $C_L$ ). Consequently, for Model 1 and Model 2, we can obtain the model score earlier in the decoding process.

## 7 Experiments

Our baseline system is a state-of-the-art string-to-dependency system (Shen et al., 2008). The system is trained on 10 million parallel sentences that are available to the Phase 1 of the DARPA BOLT Chinese-English MT task. The training corpora include a mixed genre of newswire, weblog, broadcast news, broadcast conversation, discussion forums and comes from various sources such as LDC, HK Law, HK Hansard and UN data.

In total, our baseline model employs about 40 features, including four from our proposed Two-Neighbor Orientation model. In addition to the standard features including the rule translation probabilities, we incorporate features that are found useful for developing a state-of-the-art baseline, such as the provenance features (Chiang et al., 2011). We use a large 6-gram language model, which was trained on 10 billion English words from multiple corpora, including the English side of our parallel corpus plus other corpora such as Gigaword (LDC2011T07) and Google News. We also train a class-based language model (Chen, 2009) on two million English sentences selected from the parallel corpus. As the backbone of our string-to-dependency system, we train 3-gram models for left and right dependencies and unigram for head using the target side of the bilingual training data. To train our Two-Neighbor Orientation model, we select a subset of 5 million aligned sentence pairs.

For the tuning and development sets, we set aside 1275 and 1239 sentences selected from LDC2010E30 corpus. We tune the decoding weights with PRO (Hopkins and May, 2011) to maximize BLEU-TER. As for the blind test set, we report the performance on the NIST MT08 evaluation set, which consists of 691 sentences from newswire and 666 sentences from weblog. We pick the weights that produce the highest development set scores to decode the test set.

Table 4 summarizes the experimental results on NIST MT08 newswire and weblog. In column 2, we report the classification accuracy on a subset of training data. Note that these numbers are for reference only and not directly comparable with each other since the features used in these classifiers include several gold standard information, such as the anchors’ target words, the anchors’ MOS-related features (Model 3 & 4) and the orientation of the right MOS (Model 2-4); all of which have

	Acc	MT08 nw		MT08 wb	
		BLEU	TER	BLEU	TER
S2D	-	36.77	53.28	26.34	57.41
M1	72.5	37.60	52.70	27.59	56.33
M2	77.4	37.86	52.68	27.74	56.11
M3	84.5	38.02	52.42	28.22	<b>55.82</b>
M4	84.5	<b>38.55</b>	<b>52.41</b>	<b>28.44</b>	56.45

Table 4: The NIST MT08 results on newswire (nw) and weblog (wb) genres. S2D is the baseline string-to-dependency system (line 1), on top of which Two-Neighbor Orientation Model 1 to 4 are employed (line 2-5). The best TER and BLEU results on each genre are in **bold**. For BLEU, higher scores are better, while for TER, lower scores are better.

to be predicted at decoding time.

In columns 2 and 4, we report the BLEU scores, while in columns 3 and 5, we report the TER scores. The performance of our baseline string-to-dependency syntax-based SMT is shown in the first line, followed by the performance of our Two-Neighbor Orientation model starting from Model 1 to Model 4. As shown, the empirical results confirm our intuition that SMT can greatly benefit from reordering model that incorporate cross-unit contextual information.

Model 1 provides most of the gain across the two genres of around +0.9 to +1.2 BLEU and -0.5 to -1.1 TER. Model 2 which conditions  $P_{O_L}$  on  $O_R$  provides an additional +0.2 BLEU improvement on BLEU score consistently across the two genres. As shown in line 4, we see a stronger improvement in the inclusion of MOS-related information as features in Model 3. In newswire, Model 3 gives an additional +0.4 BLEU and -0.2 TER, while in weblog, it gives a stronger improvement of an additional +0.5 BLEU and -0.3 TER. The inclusion of explicit MOS modeling in Model 4 gives a significant BLEU score improvement of +0.5 but no TER improvement in newswire. In weblog, Model 4 gives a mixed results of +0.2 BLEU score improvement and a hit of +0.6 TER. We conjecture that the weblog text has a more ambiguous orientation span that are more challenging to learn. In total, our TNO model gives an encouraging result. Our most advanced model gives significant improvement of +1.8 BLEU/-0.8 TER in newswire domain and +2.1 BLEU/-1.0 TER over a strong string-to-dependency syntax-based SMT enhanced with additional state-of-the-art features.

## 8 Related Work

Our work intersects with existing work in many different respects. In this section, we mainly focus on work related to the probabilistic conditioning of our TNO model and the MOS modeling.

Our TNO model is closely related to the Unigram Orientation Model (UOM) (Tillman, 2004), which is the *de facto* reordering model of phrase-based SMT (Koehn et al., 2007). UOM views reordering as a process of generating  $(b, o)$  in a left-to-right fashion, where  $b$  is the current phrase pair and  $o$  is the orientation of  $b$  with the previously generated phrase pair  $b'$ . UOM makes strong independence assumptions and formulates the model as  $P(o|b)$ . Tillmann and Zhang (2007) proposed a Bigram Orientation Model (BOM) to include both phrase pairs  $(b$  and  $b')$  into the model. Their original intent is to model  $P(o, b|o', b')$ , but perhaps due to sparsity concerns, they settle with  $P(o|b, b')$ , dropping the conditioning on the previous orientation  $o'$ . Subsequent improvements use the  $P(o|b, b')$  formula, for example, for incorporating various linguistics feature like part-of-speech (Zens and Ney, 2006), syntactic (Chang et al., 2009), dependency information (Bach et al., 2009) and predicate-argument structure (Xiong et al., 2012). Our TNO model is more faithful to the BOM’s original formulation.

Our MOS concept is also closely related to hierarchical reordering model (Galley and Manning, 2008) in phrase-based decoding, which computes  $o$  of  $b$  with respect to a multi-block unit that may go beyond  $b'$ . They mainly use it to avoid overestimating “discontiguous” orientation but fall short in modeling the multi-block unit, perhaps due to data sparsity issue. Our MOS is also closely related to the efforts of modeling the span of hierarchical phrases in formally syntax-based SMT. Early works reward/penalize spans that respect the syntactic parse constituents of an input sentence (Chiang, 2005), and (Marton and Resnik, 2008). (Xiong et al., 2009) learn the boundaries from parsed and aligned training data, while (Xiong et al., 2010) learn the boundaries from aligned training data alone. Recent work couples span modeling tightly with reordering decisions, either by adding an additional feature for each hierarchical phrase (Chiang et al., 2008; Shen et al., 2009) or by refining the nonterminal label (Venugopal et al., 2009; Huang et al., 2010; Zollmann and Vogel, 2011). Common to this work is that the spans

modeled may not correspond to MOS, which may be suboptimal as discussed in Sec. 3.

In equating anchors with the function word class, our work, particularly Model 1, is closely related to the function word-centered model of Setiawan et al. (2007) and Setiawan et al. (2009). However, we provide a discriminative treatment to the model to include a richer set of features including the MOS modeling. Our work in incorporating global context also intersects with existing work in Preordering Model (PM), e.g. (Niehues and Kolss, 2009; Costa-jussà and Fonollosa, 2006; Genzel, 2010; Visweswariah et al., 2011; Tromble and Eisner, 2009). The goal of PM is to reorder the input sentence  $F$  into  $F'$  whose order is closer to the target language order, whereas the goal of our model is to directly reorder  $F$  into the target language order. The crucial difference is that we have to integrate our model into SMT decoder, which is highly non-trivial.

## 9 Conclusion

We presented a novel approach to address a kind of long-distance reordering that requires global cross-boundary contextual information. Our approach, which we formulate as a Two-Neighbor Orientation model, includes the joint modeling of two orientation decisions and the modeling of the maximal span of the reordered chunks through the concept of Maximal Orientation Span. We describe four versions of the model and implement an algorithm to integrate our proposed model into a syntax-based SMT system. Empirical results confirm our intuition that incorporating cross-boundaries contextual information improves translation quality. In a large scale Chinese-to-English translation task, we achieve a significant improvement over a strong baseline. In the future, we hope to continue this line of research, perhaps by learning to identify anchors automatically from training data, incorporating a richer set of linguistics features such as dependency structure and strengthening the modeling of Maximal Orientation Span.

## Acknowledgements

We would like to acknowledge the support of DARPA under Grant HR0011-12-C-0015 for funding part of this work. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the DARPA.

## References

- Nguyen Bach, Qin Gao, and Stephan Vogel. 2009. Source-side dependency tree reordering models with subtree movements and constraints. In *Proceedings of the Twelfth Machine Translation Summit (MTSummit-XII)*, Ottawa, Canada, August. International Association for Machine Translation.
- Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. Discriminative reordering with Chinese grammatical relations features. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation (SSST-3) at NAACL HLT 2009*, pages 51–59, Boulder, Colorado, June. Association for Computational Linguistics.
- Stanley Chen. 2009. Shrinking exponential language models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 468–476, Boulder, Colorado, June. Association for Computational Linguistics.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Honolulu, Hawaii, October.
- David Chiang, Steve DeNeefe, and Michael Pust. 2011. Two easy improvements to lexical weighting. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 455–460, Portland, Oregon, USA, June. Association for Computational Linguistics.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Marta R. Costa-jussà and José A. R. Fonollosa. 2006. Statistical machine reordering. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 70–76, Sydney, Australia, July. Association for Computational Linguistics.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.
- Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Dmitriy Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 376–384, Beijing, China, August. Coling 2010 Organizing Committee.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Zhongqiang Huang, Martin Cmejrek, and Bowen Zhou. 2010. Soft syntactic constraints for hierarchical phrase-based translation using latent syntactic distributions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 138–147, Cambridge, MA, October. Association for Computational Linguistics.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation, June.
- Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of The 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1003–1011, Columbus, Ohio, June.
- Masaaki Nagata, Kuniko Saito, Kazuhide Yamamoto, and Kazuteru Ohashi. 2006. A clustered global phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 713–720, Sydney, Australia, July. Association for Computational Linguistics.
- Jan Niehues and Muntsin Kolss. 2009. A POS-based model for long-range reorderings in SMT. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 206–214, Athens, Greece, March. Association for Computational Linguistics.
- Hendra Setiawan, Min-Yen Kan, and Haizhou Li. 2007. Ordering phrases with function words. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 712–719, Prague, Czech Republic, June. Association for Computational Linguistics.
- Hendra Setiawan, Min Yen Kan, Haizhou Li, and Philip Resnik. 2009. Topological ordering of function words in hierarchical phrase-based translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*

- of the AFNLP, pages 324–332, Suntec, Singapore, August. Association for Computational Linguistics.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June. Association for Computational Linguistics.
- Libin Shen, Jinxi Xu, Bing Zhang, Spyros Matsoukas, and Ralph Weischedel. 2009. Effective use of linguistic and contextual information for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 72–80, Singapore, August. Association for Computational Linguistics.
- Christoph Tillman. 2004. A unigram orientation model for statistical machine translation. In *HLT-NAACL 2004: Short Papers*, pages 101–104, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Christoph Tillmann and Tong Zhang. 2007. A block bigram prediction model for statistical machine translation. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(3).
- Roy Tromble and Jason Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1007–1016, Singapore, August. Association for Computational Linguistics.
- Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. 2009. Preference grammars: Softening syntactic constraints to improve statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 236–244, Boulder, Colorado, June. Association for Computational Linguistics.
- Karthik Visweswariah, Rajkrishnan Rajkumar, Ankur Gandhe, Ananthkrishnan Ramanathan, and Jiri Navratil. 2011. A word reordering model for improved machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 486–496, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Deyi Xiong, Min Zhang, Aiti Aw, and Haizhou Li. 2009. A syntax-driven bracketing model for phrase-based translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 315–323, Suntec, Singapore, August. Association for Computational Linguistics.
- Deyi Xiong, Min Zhang, and Haizhou Li. 2010. Learning translation boundaries for phrase-based decoding. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 136–144, Los Angeles, California, June. Association for Computational Linguistics.
- Deyi Xiong, Min Zhang, and Haizhou Li. 2012. Modeling the translation of predicate-argument structure for smt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 902–911, Jeju Island, Korea, July. Association for Computational Linguistics.
- Richard Zens and Hermann Ney. 2006. Discriminative reordering models for statistical machine translation. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL): Proceedings of the Workshop on Statistical Machine Translation*, pages 55–63, New York City, NY, June. Association for Computational Linguistics.
- Andreas Zollmann and Stephan Vogel. 2011. A word-class approach to labeling pscfg rules for machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1–11, Portland, Oregon, USA, June. Association for Computational Linguistics.

# Cut the noise: Mutually reinforcing reordering and alignments for improved machine translation

**Karthik Visweswariah**      **Mitesh M. Khapra**      **Ananthakrishnan Ramanathan**  
IBM Research India      IBM Research India      IBM Research India  
v-karthik@in.ibm.com      mikhapra@in.ibm.com      anandr42@gmail.com

## Abstract

Preordering of a source language sentence to match target word order has proved to be useful for improving machine translation systems. Previous work has shown that a reordering model can be learned from high quality manual word alignments to improve machine translation performance. In this paper, we focus on further improving the performance of the reordering model (and thereby machine translation) by using a larger corpus of sentence aligned data for which manual word alignments are not available but automatic machine generated alignments are available. The main challenge we tackle is to generate quality data for training the reordering model in spite of the machine alignments being noisy. To mitigate the effect of noisy machine alignments, we propose a novel approach that improves reorderings produced given noisy alignments and also improves word alignments using information from the reordering model. This approach generates alignments that are 2.6 f-Measure points better than a baseline supervised aligner. The data generated allows us to train a reordering model that gives an improvement of 1.8 BLEU points on the NIST MT-08 Urdu-English evaluation set over a reordering model that only uses manual word alignments, and a gain of 5.2 BLEU points over a standard phrase-based baseline.

## 1 Introduction

Dealing with word order differences between source and target languages presents a significant challenge for machine translation systems. Failing to produce target words in the correct order results

in machine translation output that is not fluent and is often very hard to understand. These problems are particularly severe when translating between languages which have very different structure.

Phrase based systems (Koehn et al., 2003) use lexicalized distortion models (Al-Onaizan and Papineni, 2006; Tillman, 2004) and scores from the target language model to produce words in the correct order in the target language. These systems typically are only able to capture short range reorderings and the amount of data required to potentially capture longer range reordering phenomena is prohibitively large.

There has been a large body of work showing the efficacy of preordering source sentences using a source parser and applying hand written or automatically learned rules (Collins et al., 2005; Wang et al., 2007; Ramanathan et al., 2009; Xia and McCord, 2004; Genzel, 2010; Visweswariah et al., 2010). Recently, approaches that address the problem of word order differences between the source and target language without requiring a high quality source or target parser have been proposed (DeNero and Uszkoreit, 2011; Visweswariah et al., 2011; Neubig et al., 2012). These methods use a small corpus of manual word alignments (where the words in the source sentence are manually aligned to the words in the target sentence) to learn a model to preorder the source sentence to match target order.

In this paper, we build upon the approach in (Visweswariah et al., 2011) which uses manual word alignments for learning a reordering model. Specifically, we show that we can significantly improve reordering performance by using a large number of sentence pairs for which manual word alignments are not available. The motivation for going beyond manual word alignments is clear: the reordering model can have millions of features and estimating weights for the features on thousands of sentences of manual word alignments is

likely to be inadequate. One approach to deal with this problem would be to use only part-of-speech tags as features for all but the most frequent words. This will cut down on the number of features and perhaps the model would be learnable with a small set of manual word alignments. Unfortunately, as we will see in the experimental section, leaving out lexical information from the models hurts performance even with a relatively small set of manual word alignments. Another option would be to collect more manual word alignments but this is undesirable because it is time consuming and expensive.

The challenge in going beyond manual word alignments and using machine alignments is the noise in the machine alignments which affects the performance of the reordering model (see Section 5). We illustrate this with the help of a motivating example. Consider the example English sentence and its translation shown in Figure 1.

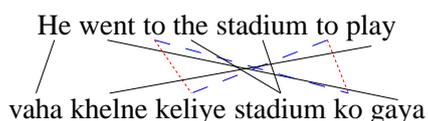


Figure 1: An example English sentence with its Urdu translation with alignment links. Red (dotted) links are incorrect links while the blue (dashed) links are the corresponding correct links.

A standard word alignment algorithm that we used (McCarley et al., 2011) made the mistake of mis-aligning the Urdu *ko* and *keliye* (it switched the two). Deriving reference reorderings from these wrong alignments would give us an incorrect reordering. A reordering model trained on such incorrect reorderings would obviously perform poorly. Our task is thus two-fold (i) improve the quality of machine alignments (ii) use these less noisy alignments to derive cleaner training data for a reordering model.

Before proceeding, we first point out that the two tasks, *viz.*, reordering and word alignment are related: Having perfect reordering makes the alignment task easier while having perfect alignments in turn makes the task of finding reorderings trivial. Motivated by this fact, we introduce models that allow us to connect the source/target reordering and the word alignments and show that these models help in mutually improving the performance of word alignments and reordering. Specifically, we build two models: the first scores

reorderings given the source sentence and noisy alignments, the second scores alignments given the noisy source and target reorderings and the source and target sentences themselves. The second model helps produce better alignments, while we use the first model to help generate better reference reordering given noisy alignments. These improved reference reorderings will then be used to train a reordering model.

Our experiments show that reordering models trained using these improved machine alignments perform significantly better than models trained only on manual word alignments. This results in a 1.8 BLEU point gain in machine translation performance on an Urdu-English machine translation task over a preordering model trained using only manual word alignments. In all, this increases the gain in performance by using the preordering model to 5.2 BLEU points over a standard phrase-based system with no preordering.

The rest of this paper is structured as follows. Section 2 describes the main reordering issues in Urdu-English translation. Section 3 introduces the reordering modeling framework that forms the basis for our work. Section 4 describes the two models we use to tie together reordering and alignments and how we use these models to generate training data for training our reordering model. Section 5 presents the experimental setup used for evaluating the models proposed in this paper on an Urdu-English machine translation task. Section 6 presents the results of our experiments. We describe related work in Section 7 and finally present some concluding remarks and potential future work in Section 8.

## 2 Reordering issues in Urdu-English translation

In this section we describe the main sources of word order differences between Urdu and English since this is the language pair we experiment with in this paper.

The typical word order in Urdu is *Subject-Object-Verb* unlike English in which the order is *Subject-Verb-Object*. Urdu has case markers that sometimes (but not always) mark the subject and the object of a sentence. This difference in the placement of verbs can often lead to movements of verbs over long distances (depending on the number of words in the object). Phrase based systems do not capture such long distance movements well.

Another difference is that Urdu uses post-positions unlike English which uses prepositions. This can also lead to long range movements depending on the length of the noun phrase that the post-position follows. The order of noun phrases and prepositional phrases is also swapped in Urdu as compared with English.

### 3 Reordering model

In this section we briefly describe the reordering model (Visweswariah et al., 2011) that forms the basis of our work. We also describe an approximation we make in the training process that significantly speeds up the training without much loss of accuracy which enables training on much larger data sets. Consider a source sentence  $\mathbf{w}$  that we would like to reorder to match the target order. Let  $\pi$  represent a candidate permutation of the source sentence  $\mathbf{w}$ .  $\pi_i$  denotes the index of the word in the source sentence that maps to position  $i$  in the candidate reordering, thus reordering with this candidate permutation  $\pi$  we will reorder the sentence  $\mathbf{w}$  to  $w_{\pi_1}, w_{\pi_2}, \dots, w_{\pi_n}$ . The reordering model we use assigns costs to candidate permutations as:

$$C(\pi|\mathbf{w}) = \sum_i c(\pi_{i-1}, \pi_i).$$

The costs  $c(m, n)$  are pairwise costs of putting  $w_m$  immediately before  $w_n$  in the reordering. We reorder the sentence  $\mathbf{w}$  according to the permutation  $\pi$  that minimizes the cost  $C(\pi|\mathbf{w})$ . We find the minimal cost permutation by converting the problem into a symmetric Travelling Salesman Problem (TSP) and then using an implementation of the chained Lin-Kernighan heuristic (Applegate et al., 2003). The costs in the reordering model  $c(m, n)$  are parameterized by a linear model:

$$c(m, n) = \theta^T \Phi(\mathbf{w}, m, n)$$

where  $\theta$  is a learned vector of weights and  $\Phi$  is a vector of binary feature functions that inspect the words and POS tags of the source sentence at and around positions  $m$  and  $n$ . We use the features ( $\Phi$ ) described in Visweswariah et al. (2011) that were based on features used in dependency parsing (McDonald et al., 2005a).

To learn the weight vector  $\theta$  we require a corpus of sentences  $\mathbf{w}$  with their desired reorderings  $\pi^*$ . Past work Visweswariah et al. (2011) used high quality manual word alignments to derive the desired reorderings  $\pi^*$  as follows. Given word

aligned source and target sentences, we drop the source words that are not aligned<sup>1</sup>. Let  $m_i$  be the mean of the target word positions that the source word at index  $i$  is aligned to. We then sort the source indices in increasing order of  $m_i$  (this order defines  $\pi^*$ ). If  $m_i = m_j$  (for example, because  $w_i$  and  $w_j$  are aligned to the same set of words) we keep them in the same order that they occurred in the source sentence.

We used the single best Margin Infused Relaxed Algorithm (MIRA) (McDonald et al. (2005b), Crammer and Singer (2003)) with online updates to our parameters given by:

$$\begin{aligned} \theta_{i+1} &= \arg \min_{\theta} \|\theta - \theta_i\| \\ \text{s.t. } & C(\pi^*|\mathbf{w}) < C(\hat{\pi}|\mathbf{w}) - L(\pi^*, \hat{\pi}). \end{aligned}$$

In the equation above,  $\hat{\pi} = \arg \min_{\pi} C(\pi|\mathbf{w})$  is the best reordering based on the current parameter value  $\theta_i$  and  $L$  is a loss function. We take  $L$  to be the number of words for which the hypothesized permutation  $\hat{\pi}$  has a different preceding word as compared with the reference permutation  $\pi^*$ .

In this paper we focus on the case where in addition to using a relatively small number of manual word aligned sentences to derive the reference permutations  $\pi^*$  used to train our model, we would like to use more abundant but noisier machine aligned sentence pairs. To handle the larger amount of training data we obtain from machine alignments, we make an approximation in training that we found empirically to not affect performance but that makes training faster by more than a factor of five. This allows us to train the reordering model with roughly 150K sentences in about two hours. The approximation we make is that instead of using the chained Lin-Kernighan heuristic to solve the TSP problem to find  $\hat{\pi} = \arg \min_{\pi} C(\pi|\mathbf{w})$ , we select greedily for each word the preceding word that has the lowest cost<sup>2</sup>. Using  $\psi_i$  to denote  $\arg \min_j c(j, i)$  and letting

$$C(\psi|\mathbf{w}) = \sum_i c(\psi_i, i),$$

<sup>1</sup>Note that the unaligned source words are dropped only at the time of training. At the time of testing all source words are retained as the alignment information is obviously not available at test time.

<sup>2</sup>It should be noted that this approximation was done only at the time of training. At the time of testing we still use the chained Lin-Kernighan heuristic to solve the TSP problem.

we do the update according to:

$$\theta_{i+1} = \underset{\theta}{\operatorname{arg\,min}} \|\theta - \theta_i\|$$

$$s.t. \quad C(\pi^*|\mathbf{w}) < C(\psi|\mathbf{w}) - L(\pi^*, \psi).$$

Again the loss  $L(\pi^*, \psi)$  is the number of positions  $i$  for which  $\pi_{i-1}^*$  is different from  $\psi_{i-1}$ .

#### 4 Generating reference reordering from parallel sentences

The main aim of our work is to improve the reordering model by using parallel sentences for which manual word alignments are not available. In other words, we want to generate relatively clean reference reorderings from parallel sentences and use them for training a reordering model. A straightforward approach for this is to use a supervised aligner to align the words in the sentences and then derive the reference reordering as we do for manual word alignments. However, as we will see in the experimental results, the quality of a reordering model trained from automatic alignments is very sensitive to the quality of alignments. This motivated us to explore if we can further improve our aligner and the method for generating reference reorderings given alignments.

We improve upon the above mentioned basic approach by coupling the tasks of reordering and word alignment. We do this by building a **reordering model** ( $C(\pi^s|\mathbf{w}^s, \mathbf{w}^t, \mathbf{a})$ ) that scores reorderings  $\pi^s$  given the source sentence  $\mathbf{w}^s$ , target sentence  $\mathbf{w}^t$  and machine alignments  $\mathbf{a}$ . Complementing this model, we build an **alignment model** ( $P(\mathbf{a}|\mathbf{w}^s, \mathbf{w}^t, \pi^s, \pi^t)$ ) that scores alignments  $\mathbf{a}$  given the source and target sentences and their predicted reorderings according to source and target reordering models. The model ( $C(\pi^s|\mathbf{w}^s, \mathbf{w}^t, \mathbf{a})$ ) helps to produce better reference reorderings for training our final reordering model given fixed machine alignments and the alignment model ( $P(\mathbf{a}|\mathbf{w}^s, \mathbf{w}^t, \pi^s, \pi^t)$ ) helps improve the machine alignments taking into account information from reordering models. In the following sections, we describe our overall approach followed by a description of the two models.

##### 4.1 Overall approach to generating training data

We first describe our overall approach to generating training data for the reordering model given a small corpus of sentences with manual

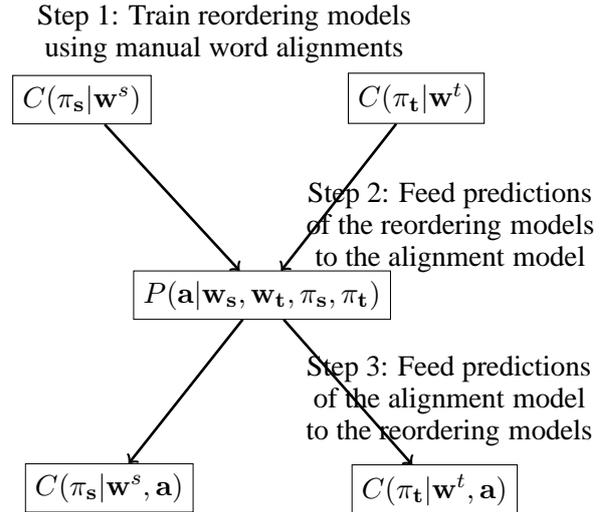


Figure 2: Overall approach: Building a sequence of reordering and alignment models.

word alignments ( $H$ ) and a much larger corpus of parallel sentences ( $U$ ) that are not word aligned. The basic idea is to chain together the two models, *viz.*, reordering model and alignment model, as illustrated in Figure 2. The steps involved are as described below:

**Step 1:** First, we use manual word alignments ( $H$ ) to train source and target reordering models as described in (Visweswariah et al., 2011).

**Step 2:** Next, we use the hand alignments to train an alignment model  $P(\mathbf{a}|\mathbf{w}^s, \mathbf{w}^t, \pi^s, \pi^t)$ . In addition to the original source and target sentence, we also feed the predictions of the reordering model trained in Step 1 to this alignment model (see section 4.2 for details of the model itself).

**Step 3:** Finally, we use the predictions of the alignment model trained in Step 2 to train reordering models  $C(\pi^s|\mathbf{w}^s, \mathbf{w}^t, \mathbf{a})$  (see section 4.3 for details on the reordering model itself).

After building the sequence of models shown in Figure 2, we apply them in sequence on the unaligned parallel data  $U$ , starting with the reordering models  $C(\pi^s|\mathbf{w}^s)$  and  $C(\pi^t|\mathbf{w}^t)$ . The reorderings obtained for the source side in  $U$  (after applying the final model  $C(\pi^s|\mathbf{w}^s, \mathbf{a})$ ) are used along with reference reorderings obtained from the manual word alignments to train our reordering model. Note that, in theory, we could iterate over steps 2 and 3 several times but, in practice we did not see a benefit of going beyond one iter-

ation in our experiments. Also, since we are interested only in the source side reorderings produced by the model  $C(\pi^s | \mathbf{w}^s, \mathbf{a})$ , the target reordering model  $C(\pi^t | \mathbf{w}^t, \mathbf{a})$  is needed only if we iterate over steps 2 and 3.

We now point to some practical considerations of our approach. Consider the case when we are training an alignment model conditioned on reorderings ( $P(\mathbf{a} | \mathbf{w}^s, \mathbf{w}^t, \pi^s, \pi^t)$ ). If the reordering model that generated these reorderings  $\pi^s, \pi^t$  were trained on the same data that we are using to train the alignment model, then the reorderings would be much better than we would expect on unseen test data, and hence the alignment model ( $P(\mathbf{a} | \mathbf{w}^s, \mathbf{w}^t, \pi^s, \pi^t)$ ) may learn to make the alignment overly consistent with the reorderings  $\pi^s$  and  $\pi^t$ . To counter this problem, we divide the training data  $H$  into  $K$  parts and at each stage we apply a model (reordering or alignment) on part  $i$  that had not seen part  $i$  in training. This ensures that the alignment model does not see very optimistic reorderings and vice versa. We now describe the individual models, viz.,  $P(\mathbf{a} | \mathbf{w}^s, \mathbf{w}^t, \pi^s, \pi^t)$  and  $C(\pi^s | \mathbf{w}^s, \mathbf{a})$ .

## 4.2 Modeling alignments given reordering

In this section we describe how we fuse information from source and target reordering models to improve word alignments.

As a base model we use the correction model for word alignments proposed by McCarley et al. (2011). This model was significantly better than the MaxEnt aligner (Ittycheriah and Roukos, 2005) and is also flexible in the sense that it allows for arbitrary features to be introduced while still keeping training and decoding tractable by using a greedy decoding algorithm that explores potential alignments in a small neighborhood of the current alignment. The model thus needs a reasonably good initial alignment to start with for which we use the MaxEnt aligner (Ittycheriah and Roukos, 2005) as in McCarley et al. (2011).

The correction model is a log-linear model:

$$P(\mathbf{a} | \mathbf{w}^s, \mathbf{w}^t) = \frac{\exp(\lambda^T \phi(\mathbf{a}, \mathbf{w}^s, \mathbf{w}^t))}{Z(\mathbf{w}^s, \mathbf{w}^t)}.$$

The  $\lambda$ s are trained using the LFBGS algorithm (Liu et al., 1989) to maximize the log-likelihood smoothed with  $L_2$  regularization. The feature functions  $\phi$  we start with are those used in McCarley et al. (2011) and include features encoding

the Model 1 probabilities between pairs of words linked in the alignment  $\mathbf{a}$ , features that inspect source and target POS tags and parses (if available) and features that inspect the alignments of adjacent words in the source and target sentence.

To incorporate information from the reordering model, we add features that use the predicted source  $\pi^s$  and target permutations  $\pi^t$ . We introduce some notation to describe these features. Let  $S_m$  and  $S_n$  be the set of indices of target words that  $w_m^s$  and  $w_n^s$  are aligned to respectively. We define the minimum signed distance (*msd*) between these two sets as:

$$msd(S_m, S_n) = i^* - j^*$$

where,  $(i^*, j^*) = \arg \min_{(i,j) \in S_m \times S_n} |i - j|$

We quantize and encode with binary features the minimum signed distance between the sets of the indices of the target words that source words adjacent in the reordering  $\pi^s$  ( $w_{\pi_i^s}^s$  and  $w_{\pi_{i+1}^s}^s$ ) are aligned to. We instantiate similar features with the roles of source and target sentences reversed. With this addition of features we use the same training and testing procedure as in McCarley et al. (2011). If the reorderings  $\pi^s$  were perfect we would learn to only allow alignments where  $w_{\pi_i^s}^s$  and  $w_{\pi_{i+1}^s}^s$  were aligned to adjacent words in the target sentence. Although the reordering model is not perfect, preferring alignments consistent with the reordering models improves the aligner.

## 4.3 Modeling reordering given alignments

To model source permutations given source ( $\mathbf{w}^s$ ) and target ( $\mathbf{w}^t$ ) sentences, and alignments ( $\mathbf{a}$ ) we reuse the reordering model framework described in Section 3 adding additional features capturing the relation between a hypothesized permutation  $\pi$  and alignments  $\mathbf{a}$ . To allow for searching via the same TSP formulation we once again assign costs to candidate permutations as:

$$C(\pi^s | \mathbf{w}^s, \mathbf{w}^t, \mathbf{a}) = \sum_i c(\pi_{i-1}, \pi_i | \mathbf{w}^s, \mathbf{a}).$$

Note that we introduce a dependence on the target sentence  $\mathbf{w}^t$  only through the alignment  $\mathbf{a}$ . Once again we parameterize the costs by a linear model:

$$c(m, n) = \theta^T \Phi(\mathbf{w}^s, \mathbf{a}, m, n).$$

For the feature functions  $\Phi$ , in addition to the features that only depend on  $\mathbf{w}^s, m, n$  (that we

use in our standard reordering model) we add binary indicator features based on  $msd(S_m, S_n)$  and  $msd(S_m, S_n)$  conjoined with  $POS(w_m^s)$  and  $POS(w_n^s)$ .

Here,  $S_m$  and  $S_n$  are the set of indices of target words that  $w_m^s$  and  $w_n^s$  are aligned to respectively. We conjoin the  $msd$  (minimum signed distance) with the POS tags to allow the model to capture the fact that the alignment error rate maybe higher for some POS tags than others (*e.g.*, we have observed verbs have a higher error rate in Urdu-English alignments).

Given these features we train the parameters  $\theta$  using the MIRA algorithm as described in Section 3. Using this model, we can find the lowest cost permutation  $C(\pi^s | \mathbf{w}^s, \mathbf{a})$  using the Lin-Kernighan heuristic as described in Section 3. This model allows us to combine features from the original reordering model along with information coming from the alignments to find source reorderings given a parallel corpus and alignments. We will see in the experimental section that this improves upon the simple heuristic for deriving reorderings described in Section 3.

## 5 Experimental setup

In this section we describe the experimental setup that we used to evaluate the models proposed in this paper. All experiments were done on Urdu-English and we evaluate reordering in two ways: Firstly, we evaluate reordering performance directly by comparing the reordered source sentence in Urdu with a reference reordering obtained from the manual word alignments using BLEU (Papineni et al., 2002) (we call this measure monolingual BLEU or mBLEU). All mBLEU results are reported on a small test set of about 400 sentences set aside from our set of sentences with manual word alignments. Additionally, we evaluate the effect of reordering on our final systems for machine translation measured using BLEU.

We use about 10K sentences (180K words) of manual word alignments which were created in house using part of the NIST MT-08 training data<sup>3</sup> to train our baseline reordering model and to train our supervised machine aligners. We use a parallel corpus of 3.9M words consisting of 1.7M words from the NIST MT-08 training data set and 2.2M words extracted from parallel news stories on the

<sup>3</sup><http://www ldc.upenn.edu>

web<sup>4</sup>. The parallel corpus is used for building our phrased based machine translation system and to add training data for our reordering model. For our English language model, we use the Gigaword English corpus in addition to the English side of our parallel corpus. Our Part-of-Speech tagger is a Maximum Entropy Markov model tagger trained on roughly fifty thousand words from the CRULP corpus (Hussain, 2008).

For our machine translation experiments, we used a standard phrase based system (Al-Onaizan and Papineni, 2006) with a lexicalized distortion model with a window size of +/-4 words<sup>5</sup>. To extract phrases we use HMM alignments along with higher quality alignments from a supervised aligner (McCarley et al., 2011). We report results on the (four reference) NIST MT-08 evaluation set in Table 4 for the News and Web conditions. The News and Web conditions each contain roughly 20K words in the test set, with the Web condition containing more informal text from the web.

## 6 Results and Discussions

We now discuss the results of our experiments.

**Need for additional data:** We first show the need for additional data in Urdu-English reordering. Column 2 of Table 1 shows mBLEU as a function of the number of sentences with manual word alignments that are used to train the reordering model. We see a roughly 3 mBLEU points drop in performance per halving of data indicating a potential for improvement by adding more data.

**Using fewer features:** We compare the performance of a model trained using lexical features for all words (Column 2 of Table 1) with a model trained using lexical features only for the 1000 most frequent words (Column 3 of Table 1). The motivation for this is to explore if a good model can be learned even from a small amount of data if we restrict the number of features in a reasonable manner. However, we see that even with only 2.4K sentences with manual word alignments our model benefits from lexical identities of more than the 1000 most frequent words.

**Effect of quality of machine alignments:** We next look at the use of automatically generated

<sup>4</sup><http://centralasiaonline.com>

<sup>5</sup>Note that the same window size of +/-4 words was used for all the systems, *i.e.*, the baseline system as well as the systems using different preordering techniques.

Data size	All features	Frequent lex only
10K	52.5	50.8
5K	49.6	49.0
2.5K	46.6	46.2

Table 1: mBLEU scores for Urdu to English re-ordering using different number of sentences of manually word aligned training data with all features and with lexical features instantiated only for the 1000 most frequent words.

machine alignments to train the reordering model and see the effect of aligner quality on the reordering model generated using this data. These experiments also form the baseline for the models we propose in this paper to clean up alignments. We experimented with two different supervised aligners : a maximum entropy aligner (Ittycheriah and Roukos, 2005) and an improved correction model that corrects the maximum entropy alignments (McCarley et al., 2011).

Aligner		Train size (words)	mBLEU
Type	f-Measure		
None		-	35.5
Manual		180K	52.5
MaxEnt	70.0	3.9M	49.5
Correction model	78.1	3.9M	55.1

Table 2: mBLEU scores for Urdu to English re-ordering using models trained on different data sources and tested on a development set of 8017 Urdu tokens.

Table 2 shows mBLEU scores when the reordering model is trained on reordering references created from aligners with different quality. We see that the quality of the alignments matter a great deal to the reordering model; using MaxEnt alignments cause a degradation in performance over just using a small set of manual word alignments. The alignments obtained using the aligner of McCarley et al. (2011) are of much better quality and hence give higher reordering performance. Note that this reordering performance is much better than that obtained using manual word alignments because the size of machine alignments is much larger (3.9M v/s 180K words).

**Improvements in reordering performance using the proposed models:** Table 3 shows improvements in the reordering model when using the models proposed in this paper. We use  $H$  to refer to the manually word aligned data and  $U$  to refer to the additional sentence pairs for which manual word alignments are not available. We report

the following numbers :

**1. Base correction model:** This is the baseline where we use the correction model of McCarley et al. (2011) for generating word alignments. The f-Measure of this aligner is 78.1% (see row 1, column 2). Corresponding to this, we also report the baseline for our reordering experiments in the third column. Here, we first generate word alignments for  $U$  using the aligner of McCarley et al. (2011) and then extract reference reorderings from these alignments. We then combine these reference reorderings with the reference reorderings derived from  $H$  and use this combined data to train a reordering model which serves as the baseline (mBLEU = 55.1).

**2. Correction model,  $C(\pi|a)$ :** Here, once again we generate alignments for  $U$  using the correction model of McCarley et al. (2011). However, instead of using the basic approach of extracting reference reorderings, we use our improved model  $C(\pi|a)$  to generate reference reorderings from  $U$ . These reference reorderings are again combined with the reference reorderings derived from  $H$  and used to train a reordering model (mBLEU = 56.4).

**3.  $P(a|\pi)$ ,  $C(\pi|a)$ :** Here, we build the entire sequence of models shown in Figure 2. The alignment model  $P(a|\pi)$  is first improved by using predictions from the reordering model. These improved alignments are then used to extract better reference reorderings from  $U$  using  $C(\pi|a)$ .

We see substantial improvements over simply adding in the data from the machine alignments. Improvements come roughly in equal parts from the two techniques we proposed in this paper : (i) using a model to generate reference reorderings from noisy alignments and (ii) using reordering information to improve the aligner.

Method	f-Measure	mBLEU
Base Correction model	78.1	55.1
Correction model, $C(\pi a)$	78.1	56.4
$P(a \pi)$ , $C(\pi a)$	80.7	57.6

Table 3: mBLEU with different methods to generate reordering model training data from a machine aligned parallel corpus in addition to manual word alignments.

**Improvements in MT performance using the proposed models:** We report results for a phrase based system with different preordering techniques. For results including a reordering model, we simply reorder the source side Urdu data both while training and at test time. In addition to

phrase based systems with different reordering methods, we also report on a hierarchical phrase based system for which we used Joshua 4.0 (Ganitkevitch et al., 2012). We see a significant gain of 1.8 BLEU points in machine translation by going beyond manual word alignments using the best reordering model reported in Table 3. We also note a gain of 2.0 BLEU points over a hierarchical phrase based system.

System type	MT-08 eval		
	Web	News	All
Baseline (no reordering)	18.4	25.6	22.2
Hierarchical phrase based	19.6	30.7	25.4
Reordering: Manual alignments	20.7	30.0	25.6
+ Machine alignments simple	21.3	30.9	26.4
+ machine alignments, model based	<b>22.1</b>	<b>32.2</b>	<b>27.4</b>

Table 4: MT performance without reordering (phrase based and hierarchical phrase based), and with reordering models using different data sources (phrase based).

## 7 Related work

Dealing with the problem of handling word order differences in machine translation has recently received much attention. The approaches proposed for solving this problem can be broadly divided into 3 sets as discussed below.

The first set of approaches handle the reordering problem as part of the decoding process. Hierarchical models (Chiang, 2007) and syntax based models (Yamada and Knight, 2002; Galley et al., 2006; Liu et al., 2006; Zollmann and Venugopal, 2006) improve upon the simpler phrase based models but with significant additional computational cost (compared with phrase based systems) due to the inclusion of chart based parsing in the decoding process. Syntax based models also require a high quality source or target language parser.

The second set of approaches rely on a source language parser and treat reordering as a separate process that is applied on the source language sentence at training and test time before using a standard approach to machine translation. Reordering the source data with hand written or automatically learned rules is effective and efficient (Collins et al., 2005; Wang et al., 2007; Ramanathan et al., 2009; Xia and McCord, 2004; Genzel, 2010; Visweswariah et al., 2010) but requires a source language parser.

Recent approaches that avoid the need for a

source or target language parser and retain the efficiency of reordering models were proposed in (Tromble and Eisner, 2009; DeNero and Uszkoreit, 2011; Visweswariah et al., 2011; Neubig et al., 2012). (DeNero and Uszkoreit, 2011; Visweswariah et al., 2011; Neubig et al., 2012) focus on the use of manual word alignments to learn reordering models and in both cases no benefit was obtained by using the parallel corpus in addition to manual word alignments. Our work is an extension of Visweswariah et al. (2011) and we focus on being able to incorporate relatively noisy machine alignments to improve the reordering model.

In addition to being related to work in reordering, our work is also more broadly related to several other efforts which we now outline. Setiawan et al. (2010) proposed the use of function word reordering to improve alignments. While this work is similar to one of our models (model of alignments given reordering) we differ in using a reordering model of all words (not just function words) and both source and target sentences (not just the source sentence). The task of directly learning a reordering model for language pairs that are very different is closely related to the task of parsing and hence work on semi-supervised parsing (Koo et al., 2008; McClosky et al., 2006; Suzuki et al., 2009) is broadly related to our work. Our work coupling reordering and alignments is also similar in spirit to approaches where parsing and alignment are coupled (Wu, 1997).

## 8 Conclusion

In the paper we showed that a reordering model can benefit from data beyond a relatively small corpus of manual word alignments. We proposed a model that scores reorderings given alignments and the source sentence that we use to generate cleaner training data from noisy alignments. We also proposed a model that scores alignments given source and target sentence reorderings that improves a supervised alignment model by 2.6 points in f-Measure. While the improvement in alignment performance is modest, the improvement does result in improved reordering models. Cumulatively, we see a gain of 1.8 BLEU points over a baseline reordering model that only uses manual word alignments, a gain of 2.0 BLEU points over a hierarchical phrase based system, and a gain of 5.2 BLEU points over a phrase based

system that uses no source reordering on a publicly available Urdu-English test set.

As future work we would like to evaluate our models on other language pairs. Another avenue of future work we would like to explore is the use of monolingual source and target data to further assist the reordering model. We hope to be able to learn lexical information such as how many arguments a verb takes, what nouns are potential subjects for a given verb by gathering statistics from an English parser and projecting to the source language via our word/phrase translation table.

## References

- Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of ACL, ACL-44*, pages 529–536, Morristown, NJ, USA. Association for Computational Linguistics.
- David Applegate, William Cook, and Andre Rohe. 2003. Chained lin-kernighan for large traveling salesman problems. In *INFORMS Journal On Computing*.
- David Chiang. 2007. Hierarchical phrase-based translation. *Comput. Linguist.*, 33(2):201–228, June.
- Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL*, pages 531–540, Morristown, NJ, USA. Association for Computational Linguistics.
- Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3:951–991, March.
- John DeNero and Jakob Uszkoreit. 2011. Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pages 193–203, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, ACL-44*, pages 961–968, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Juri Ganitkevitch, Yuan Cao, Jonathan Weese, Matt Post, and Chris Callison-Burch. 2012. Joshua 4.0: Packing, pro, and paraphrases. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 283–291, Montréal, Canada, June. Association for Computational Linguistics.
- Dmitriy Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*.
- Sarmad Hussain. 2008. Resources for Urdu language processing. In *Proceedings of the 6th Workshop on Asian Language Resources, IJCNLP'08*.
- Abraham Ittycheriah and Salim Roukos. 2005. A maximum entropy word aligner for Arabic-English machine translation. In *Proceedings of HLT/EMNLP, HLT '05*, pages 89–96, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL*.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *ACL*, pages 595–603.
- Dong C. Liu, Jorge Nocedal, and Dong C. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, ACL-44*, pages 609–616, Stroudsburg, PA, USA. Association for Computational Linguistics.
- J. Scott McCarley, Abraham Ittycheriah, Salim Roukos, Bing Xiang, and Jian-ming Xu. 2011. A correction model for word alignments. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pages 889–898, Stroudsburg, PA, USA. Association for Computational Linguistics.
- David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *HLT-NAACL*.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05*, pages 91–98, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT*.
- Graham Neubig, Taro Watanabe, and Shinsuke Mori. 2012. Inducing a discriminative parser to optimize machine translation reordering. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational*

- Natural Language Learning*, pages 843–853, Jeju Island, Korea, July. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ananthkrishnan Ramanathan, Hansraj Choudhary, Avishek Ghosh, and Pushpak Bhattacharyya. 2009. Case markers and morphology: addressing the crux of the fluency problem in English-Hindi smt. In *Proceedings of ACL-IJCNLP*.
- Hendra Setiawan, Chris Dyer, and Philip Resnik. 2010. Discriminative word alignment with a function word reordering model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 534–544, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 551–560, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Christoph Tillman. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of HLT-NAACL*.
- Roy Tromble and Jason Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of EMNLP*.
- Karthik Visweswariah, Jiri Navratil, Jeffrey Sorensen, Vijil Chenthamarakshan, and Nandakishore Kambhatla. 2010. Syntax based reordering with automatically derived rules for improved statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*.
- Karthik Visweswariah, Rajkrishnan Rajkumar, Ankur Gandhe, Ananthkrishnan Ramanathan, and Jiri Navratil. 2011. A word reordering model for improved machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 486–496, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of EMNLP-CoNLL*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Comput. Linguist.*, 23(3):377–403, September.
- Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *COLING*.
- Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical MT. In *Proceedings of ACL*.
- Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*.

# Vector Space Model for Adaptation in Statistical Machine Translation

Boxing Chen, Roland Kuhn and George Foster

National Research Council Canada

first.last@nrc-cnrc.gc.ca

## Abstract

This paper proposes a new approach to domain adaptation in statistical machine translation (SMT) based on a vector space model (VSM). The general idea is first to create a vector profile for the in-domain development (“dev”) set. This profile might, for instance, be a vector with a dimensionality equal to the number of training subcorpora; each entry in the vector reflects the contribution of a particular subcorpus to all the phrase pairs that can be extracted from the dev set. Then, for each phrase pair extracted from the training data, we create a vector with features defined in the same way, and calculate its similarity score with the vector representing the dev set. Thus, we obtain a decoding feature whose value represents the phrase pair’s closeness to the dev. This is a simple, computationally cheap form of instance weighting for phrase pairs. Experiments on large scale NIST evaluation data show improvements over strong baselines: +1.8 BLEU on Arabic to English and +1.4 BLEU on Chinese to English over a non-adapted baseline, and significant improvements in most circumstances over baselines with linear mixture model adaptation. An informal analysis suggests that VSM adaptation may help in making a good choice among words with the same meaning, on the basis of style and genre.

## 1 Introduction

The translation models of a statistical machine translation (SMT) system are trained on parallel data. Usage of language and therefore the best translation practice differs widely across genres, topics, and dialects, and even depends on a partic-

ular author’s or publication’s style; the word “domain” is often used to indicate a particular combination of all these factors. Unless there is a perfect match between the training data domain and the (test) domain in which the SMT system will be used, one can often get better performance by adapting the system to the test domain.

Domain adaptation is an active topic in the natural language processing (NLP) research community. Its application to SMT systems has recently received considerable attention. Approaches that have been tried for SMT model adaptation include mixture models, transductive learning, data selection, instance weighting, and phrase sense disambiguation, etc.

Research on mixture models has considered both linear and log-linear mixtures. Both were studied in (Foster and Kuhn, 2007), which concluded that the best approach was to combine sub-models of the same type (for instance, several different TMs or several different LMs) linearly, while combining models of different types (for instance, a mixture TM with a mixture LM) log-linearly. (Koehn and Schroeder, 2007), instead, opted for combining the sub-models directly in the SMT log-linear framework.

In transductive learning, an MT system trained on general domain data is used to translate in-domain monolingual data. The resulting bilingual sentence pairs are then used as additional training data (Ueffing et al., 2007; Chen et al., 2008; Schwenk, 2008; Bertoldi and Federico, 2009).

Data selection approaches (Zhao et al., 2004; Hildebrand et al., 2005; Lü et al., 2007; Moore and Lewis, 2010; Axelrod et al., 2011) search for bilingual sentence pairs that are similar to the in-domain “dev” data, then add them to the training data.

Instance weighting approaches (Matsoukas et al., 2009; Foster et al., 2010; Huang and Xiang, 2010; Phillips and Brown, 2011; Sennrich, 2012)

typically use a rich feature set to decide on weights for the training data, at the sentence or phrase pair level. For example, a sentence from a subcorpus whose domain is far from that of the dev set would typically receive a low weight, but sentences in this subcorpus that appear to be of a general nature might receive higher weights.

The 2012 JHU workshop on Domain Adaptation for MT <sup>1</sup> proposed phrase sense disambiguation (PSD) for translation model adaptation. In this approach, the context of a phrase helps the system to find the appropriate translation.

In this paper, we propose a new instance weighting approach to domain adaptation based on a vector space model (VSM). As in (Foster et al., 2010), this approach works at the level of phrase pairs. However, the VSM approach is simpler and more straightforward. Instead of using word-based features and a computationally expensive training procedure, we capture the distributional properties of each phrase pair directly, representing it as a vector in a space which also contains a representation of the dev set. The similarity between a given phrase pair’s vector and the dev set vector becomes a feature for the decoder. It rewards phrase pairs that are in some sense closer to those found in the dev set, and punishes the rest. In initial experiments, we tried three different similarity functions: Bhattacharyya coefficient, Jensen-Shannon divergency, and cosine measure. They all enabled VSM adaptation to beat the non-adaptive baseline, but Bhattacharyya similarity worked best, so we adopted it for the remaining experiments.

The vector space used by VSM adaptation can be defined in various ways. In the experiments described below, we chose a definition that measures the contribution (to counts of a given phrase pair, or to counts of all phrase pairs in the dev set) of each training subcorpus. Thus, the variant of VSM adaptation tested here bears a superficial resemblance to domain adaptation based on mixture models for TMs, as in (Foster and Kuhn, 2007), in that both approaches rely on information about the subcorpora from which the data originate. However, a key difference is that in this paper we explicitly capture each phrase pair’s distribution across subcorpora, and compare it to the aggregated distribution of phrase pairs in the dev set. In mixture models, a phrase pair’s distribu-

tion across subcorpora is captured only implicitly, by probabilities that reflect the prevalence of the pair within each subcorpus. Thus, VSM adaptation occurs at a much finer granularity than mixture model adaptation. More fundamentally, there is nothing about the VSM idea that obliges us to define the vector space in terms of subcorpora.

For instance, we could cluster the words in the source language into  $S$  clusters, and the words in the target language into  $T$  clusters. Then, treating the dev set and each phrase pair as a pair of bags of words (a source bag and a target bag) one could represent each as a vector of dimension  $S + T$ , with entries calculated from the counts associated with the  $S + T$  clusters (in a way similar to that described for phrase pairs below). The (dev, phrase pair) similarity would then be independent of the subcorpora. One can think of several other ways of defining the vector space that might yield even better results than those reported here. Thus, VSM adaptation is not limited to the variant of it that we tested in our experiments.

## 2 Vector space model adaptation

Vector space models (VSMs) have been widely applied in many information retrieval and natural language processing applications. For instance, to compute the sense similarity between terms, many researchers extract features for each term from its context in a corpus, define a VSM and then apply similarity functions (Hindle, 1990; Lund and Burgess, 1996; Lin, 1998; Turney, 2001).

In our experiments, we exploited the fact that the training data come from a set of subcorpora. For instance, the Chinese-English training data are made up of 14 subcorpora (see section 3 below). Suppose we have  $C$  subcorpora. The domain vector for a phrase-pair  $(f, e)$  is defined as

$$V(f, e) = \langle w_1(f, e), \dots, w_i(f, e), \dots, w_C(f, e) \rangle, \quad (1)$$

where  $w_i(f, e)$  is a standard *tf · idf* weight, i.e.

$$w_i(f, e) = tf_i(f, e) \cdot idf_i(f, e). \quad (2)$$

To avoid a bias towards longer corpora, we normalize the raw joint count  $c_i(f, e)$  in the corpus  $s_i$  by dividing by the maximum raw count of any phrase pair extracted in the corpus  $s_i$ . Let

<sup>1</sup><http://www.clsp.jhu.edu/workshops/archive/ws-12/groups/dasmt>

$$tf_i(f, e) = \frac{c_i(f, e)}{\max\{c_i(f_j, e_k), (f_j, e_k) \in s_i\}}. \quad (3)$$

The  $idf(f, e)$  is the inverse document frequency: a measure of whether the phrase-pair  $(f, e)$  is common or rare across all subcorpora. We use the standard formula:

$$idf(f, e) = \log\left(\frac{C}{df(f, e)} + \lambda\right), \quad (4)$$

where  $df(f, e)$  is the number of subcorpora that  $(f, e)$  appears in, and  $\lambda$  is an empirically determined smoothing term.

For the in-domain dev set, we first run word alignment and phrases extracting in the usual way for the dev set, then sum the distribution of each phrase pair  $(f_j, e_k)$  extracted from the dev data across subcorpora to represent its domain information. The dev vector is thus

$$V(dev) = \langle w_1(dev), \dots, w_C(dev) \rangle, \quad (5)$$

where

$$w_i(dev) = \sum_{j=0}^{j=J} \sum_{k=0}^{k=K} c_{dev}(f_j, e_k) w_i(f_j, e_k) \quad (6)$$

$J, K$  are the total numbers of source/target phrases extracted from the dev data respectively.  $c_{dev}(f_j, e_k)$  is the joint count of phrase pair  $f_j, e_k$  found in the dev set.

The vector can also be built with other features of the phrase pair. For instance, we could replace the raw joint count  $c_i(f, e)$  in Equation 3 with the raw marginal count of phrase pairs  $(f, e)$ . Therefore, even within the variant of VSM adaptation we focus on in this paper, where the definition of the vector space is based on the existence of subcorpora, one could utilize other definitions of the vectors of the similarity function than those we utilized in our experiments.

## 2.1 Vector similarity functions

VSM uses the similarity score between the vector representing the in-domain dev set and the vector representing each phrase pair as a decoder feature. There are many similarity functions we could have employed for this purpose (Cha, 2007). We

tested three commonly-used functions: the Bhattacharyya coefficient (BC) (Bhattacharyya, 1943; Kazama et al., 2010), the Jensen-Shannon divergence (JSD), and the cosine measure. According to (Cha, 2007), these belong to three different families of similarity functions: the Fidelity family, the Shannon's entropy family, and the inner Product family respectively. It was BC similarity that yielded the best performance, and that we ended up using in subsequent experiments.

To map the BC score onto a range from 0 to 1, we first normalize each weight in the vector by dividing it by the sum of the weights. Thus, we get the probability distribution of a phrase pair or the phrase pairs in the dev data across all subcorpora:

$$p_i(f, e) = \frac{w_i(f, e)}{\sum_{j=1}^{j=C} w_j(f, e)} \quad (7)$$

$$p_i(dev) = \frac{w_i(dev)}{\sum_{j=1}^{j=C} w_j(dev)} \quad (8)$$

To further improve the similarity score, we apply absolute discounting smoothing when calculating the probability distributions  $p_i(f, e)$ . We subtract a discounting value  $\alpha$  from the non-zero  $p_i(f, e)$ , and equally allocate the remaining probability mass to the zero probabilities. We carry out the same smoothing for the probability distributions  $p_i(dev)$ . The smoothing constant  $\alpha$  is determined empirically on held-out data.

The Bhattacharyya coefficient (BC) is defined as follows:

$$BC(dev; f, e) = \sum_{i=0}^{i=C} \sqrt{p_i(dev) \cdot p_i(f, e)} \quad (9)$$

The other two similarity functions we also tested are JSD and cosine (Cos). They are defined as follows:

$$JSD(dev; f, e) = \quad (10)$$

$$\frac{1}{2} \left[ \sum_{i=1}^{i=C} p_i(dev) \log \frac{2p_i(dev)}{p_i(dev) + p_i(f, e)} + \sum_{i=1}^{i=C} p_i(f, e) \log \frac{2p_i(f, e)}{p_i(dev) + p_i(f, e)} \right]$$

$$Cos(dev; f, e) = \frac{\sum_i p_i(dev) \cdot p_i(f, e)}{\sqrt{\sum_i p_i^2(dev)} \sqrt{\sum_i p_i^2(f, e)}} \quad (11)$$

corpus	# segs	# en tok	%	genres
fbis	250K	10.5M	3.7	nw
financial	90K	2.5M	0.9	fin
gale_bc	79K	1.3M	0.5	bc
gale_bn	75K	1.8M	0.6	bn ng
gale_nw	25K	696K	0.2	nw
gale_wl	24K	596K	0.2	wl
hkh	1.3M	39.5M	14.0	hans
hkl	400K	9.3M	3.3	legal
hkn	702K	16.6M	5.9	nw
isi	558K	18.0M	6.4	nw
lex&ne	1.3M	2.0M	0.7	lex
other_nw	146K	5.2M	1.8	nw
sinorama	282K	10.0M	3.5	nw
un	5.0M	164M	58.2	un
TOTAL	10.1M	283M	100.0	(all)
devtest				
tune	1,506	161K		nw wl
NIST06	1,664	189K		nw bng
NIST08	1,357	164K		nw wl

Table 1: NIST Chinese-English data. In the *genres* column: nw=newswire, bc=broadcast conversation, bn=broadcast news, wl=weblog, ng=newsgroup, un=UN proc., bng = bn & ng.

### 3 Experiments

#### 3.1 Data setting

We carried out experiments in two different settings, both involving data from NIST Open MT 2012.<sup>2</sup> The first setting is based on data from the Chinese to English constrained track, comprising about 283 million English running words. We manually grouped the training data into 14 corpora according to genre and origin. Table 1 summarizes information about the training, development and test sets; we show the sizes of the training subcorpora in number of words as a percentage of all training data. Most training subcorpora consist of parallel sentence pairs. The *isi* and *lex&ne* corpora are exceptions: the former is extracted from comparable data, while the latter is a lexicon that includes many named entities. The development set (*tune*) was taken from the NIST 2005 evaluation set, augmented with some web-genre material reserved from other NIST corpora.

The second setting uses NIST 2012 Arabic to English data, but excludes the UN data. There are about 47.8 million English running words in these

<sup>2</sup><http://www.nist.gov/itl/iad/mig/openmt12.cfm>

corpus	# segs	# en toks	%	gen
gale_bc	57K	1.6M	3.3	bc
gale_bn	45K	1.2M	2.5	bn
gale_ng	21K	491K	1.0	ng
gale_nw	17K	659K	1.4	nw
gale_wl	24K	590K	1.2	wl
isi	1,124K	34.7M	72.6	nw
other_nw	224K	8.7M	18.2	nw
TOTAL	1,512K	47.8M	100.0	(all)
devtest				
NIST06	1,664	202K		nwl
NIST08	1,360	205K		nwl
NIST09	1,313	187K		nwl

Table 2: NIST Arabic-English data. In the *gen* (genres) column: nw=newswire, bc=broadcast conversation, bn=broadcast news, ng=newsgroup, wl=weblog, nwl = nw & wl.

training data. We manually grouped the training data into 7 groups according to genre and origin. Table 2 summarizes information about the training, development and test sets. Note that for this language pair, the comparable *isi* data represent a large proportion of the training data: 72% of the English words. We use the evaluation sets from NIST 2006, 2008, and 2009 as our development set and two test sets, respectively.

#### 3.2 System

Experiments were carried out with an in-house phrase-based system similar to Moses (Koehn et al., 2007). Each corpus was word-aligned using IBM2, HMM, and IBM4 models, and the phrase table was the union of phrase pairs extracted from these separate alignments, with a length limit of 7. The translation model (TM) was smoothed in both directions with KN smoothing (Chen et al., 2011). We use the hierarchical lexicalized reordering model (RM) (Galley and Manning, 2008), with a distortion limit of 7. Other features include lexical weighting in both directions, word count, a distance-based RM, a 4-gram LM trained on the target side of the parallel data, and a 6-gram English *Gigaword* LM. The system was tuned with batch lattice MIRA (Cherry and Foster, 2012).

#### 3.3 Results

For the baseline, we simply concatenate all training data. We have also compared our approach to two widely used TM domain adaptation ap-

proaches. One is the log-linear combination of TMs trained on each subcorpus (Koehn and Schroeder, 2007), with weights of each model tuned under minimal error rate training using MIRA. The other is a linear combination of TMs trained on each subcorpus, with the weights of each model learned with an EM algorithm to maximize the likelihood of joint empirical phrase pair counts for in-domain dev data. For details, refer to (Foster and Kuhn, 2007).

The value of  $\lambda$  and  $\alpha$  (see Eq 4 and Section 2.1) are determined by the performance on the dev set of the Arabic-to-English system. For both Arabic-to-English and Chinese-to-English experiment, these values obtained on Arabic dev were used to obtain the results below:  $\lambda$  was set to 8, and  $\alpha$  was set to 0.01. (Later, we ran an experiment on Chinese-to-English with  $\lambda$  and  $\alpha$  tuned specifically for that language pair, but the performance for the Chinese-English system only improved by a tiny, insignificant amount).

Our metric is case-insensitive IBM BLEU (Papineni et al., 2002), which performs matching of n-grams up to  $n = 4$ ; we report BLEU scores averaged across both test sets NIST06 and NIST08 for Chinese; NIST08 and NIST09 for Arabic. Following (Koehn, 2004), we use the bootstrap-resampling test to do significance testing. In tables 3 to 5, \* and \*\* denote significant gains over the baseline at  $p < 0.05$  and  $p < 0.01$  levels, respectively.

We first compare the performance of different similarity functions: cosine (COS), Jensen-Shannon divergence (JSD) and Bhattacharyya coefficient (BC). The results are shown in Table 3. All three functions obtained improvements. Both COS and BC yield statistically significant improvements over the baseline, with BC performing better than COS by a further statistically significant margin. The Bhattacharyya coefficient is explicitly designed to measure the overlap between the probability distributions of two statistical samples or populations, which is precisely what we are trying to do here: we are trying to reward phrase pairs whose distribution is similar to that of the dev set. Thus, its superior performance in these experiments is not unexpected.

In the next set of experiments, we compared VSM adaptation using the BC similarity function with the baseline which concatenates all training data and with log-linear and linear TM mixtures

system	Chinese	Arabic
baseline	31.7	46.8
COS	32.3*	47.8**
JSD	32.1	47.1
BC	33.0**	48.4**

Table 3: Comparison of different similarity functions. \* and \*\* denote significant gains over the baseline at  $p < 0.05$  and  $p < 0.01$  levels, respectively.

system	Chinese	Arabic
baseline	31.7	46.8
loglinear tm	28.4	44.5
linear tm	32.7**	47.5**
<b>vsm, BC</b>	<b>33.0**</b>	<b>48.4**</b>

Table 4: Results for variants of adaptation.

whose components are based on subcorpora. Table 4 shows that log-linear combination performs worse than the baseline: the tuning algorithm failed to optimize the log-linear combination even on dev set. For Chinese, the BLEU score of the dev set on the baseline system is 27.3, while on the log-linear combination system, it is 24.0; for Arabic, the BLEU score of the dev set on the baseline system is 46.8, while on the log-linear combination system, it is 45.4. We also tried adding the global model to the loglinear combination and it didn't improve over the baseline for either language pair. Linear mixture was significantly better than the baseline at the  $p < 0.01$  level for both language pairs. Since our approach, VSM, performed better than the linear mixture for both pairs, it is of course also significantly better than the baseline at the  $p < 0.01$  level.

This raises the question: is VSM performance significantly better than that of a linear mixture of TMs? The answer (not shown in the table) is that for Arabic to English, VSM performance is better than linear mixture at the  $p < 0.01$  level. For Chinese to English, the argument for the superiority of VSM over linear mixture is less convincing: there is significance at the  $p < 0.05$  for one of the two test sets (NIST06) but not for the other (NIST08). At any rate, these results establish that VSM adaptation is clearly superior to linear mixture TM adaptation, for one of the two language pairs.

In Table 4, the VSM results are based on the

system	Chinese	Arabic
baseline	31.7	46.8
linear tm	32.7**	47.5**
vsm, joint	33.0**	48.4**
vsm, src-marginal	32.2*	47.3*
vsm, tgt-marginal	32.6**	47.6**
vsm, src+tgt (2 feat.)	32.7**	48.2**
vsm, joint+src (2 feat.)	32.9**	48.4**
vsm, joint+tgt (2 feat.)	32.9**	48.4**
<b>vsm, joint+src+tgt (3 feat.)</b>	<b>33.1**</b>	<b>48.6**</b>

Table 5: Results for adaptation based on joint or marginal counts.

vector of the joint counts of the phrase pair. In the next experiment, we replace the joint counts with the source or target marginal counts. In Table 5, we first show the results based on source and target marginal counts, then the results of using feature sets drawn from three decoder VSM features: a joint count feature, a source marginal count feature, and a target marginal count feature. For instance, the last row shows the results when all three features are used (with their weights tuned by MIRA). It looks as though the source and target marginal counts contain useful information. The best performance is obtained by combining all three sources of information. The 3-feature version of VSM yields +1.8 BLEU over the baseline for Arabic to English, and +1.4 BLEU for Chinese to English.

When we compared two sets of results in Table 4, the joint count version of VSM and linear mixture of TMs, we found that for Arabic to English, VSM performance is better than linear mixture at the  $p < 0.01$  level; the Chinese to English significance test was inconclusive (VSM found to be superior to linear mixture at  $p < 0.05$  for NIST06 but not for NIST08). We now have somewhat better results for the 3-feature version of VSM shown in Table 5. How do these new results affect the VSM vs. linear mixture comparison? Naturally, the conclusions for Arabic don't change. For Chinese, 3-feature VSM is now superior to linear mixture at  $p < 0.01$  on NIST06 test set, but 3-feature VSM still doesn't have a statistically significant edge over linear mixture on NIST08 test set. A fair summary would be that 3-feature VSM adaptation is decisively superior to linear mixture adaptation for Arabic to English, and highly competitive with linear mixture adap-

tation for Chinese to English.

Our last set of experiments examined the question: when added to a system that already has some form of linear mixture model adaptation, does VSM improve performance? In (Foster and Kuhn, 2007), two kinds of linear mixture were described: linear mixture of language models (LMs), and linear mixture of translation models (TMs). Some of the results reported above involved linear TM mixtures, but none of them involved linear LM mixtures. Table 6 shows the results of different combinations of VSM and mixture models. \* and \*\* denote significant gains over the row *no vsm* at  $p < 0.05$  and  $p < 0.01$  levels, respectively. This means that in the table, the baseline within each box containing three results is the topmost result in the box. For instance, with an initial Chinese system that employs linear mixture LM adaptation (lin-lm) and has a BLEU of 32.1, adding 1-feature VSM adaptation (+vsm, joint) improves performance to 33.1 (improvement significant at  $p < 0.01$ ), while adding 3-feature VSM instead (+vsm, 3 feat.) improves performance to 33.2 (also significant at  $p < 0.01$ ). For Arabic, including either form of VSM adaptation always improves performance with significance at  $p < 0.01$ , even over a system including both linear TM and linear LM adaptation. For Chinese, adding VSM still always yields an improvement, but the improvement is not significant if linear TM adaptation is already in the system. These results show that combining VSM adaptation and either or both kinds of linear mixture adaptation never hurts performance, and often improves it by a significant amount.

### 3.4 Informal Data Analysis

To get an intuition for how VSM adaptation improves BLEU scores, we compared outputs from the baseline and VSM-adapted system (“vsm, joint” in Table 5) on the Chinese test data. We focused on examples where the two systems had translated the same source-language (Chinese) phrase  $s$  differently, and where the target-language (English) translation of  $s$  chosen by the VSM-adapted system,  $t_V$ , had a higher Bhattacharyya score for similarity with the dev set than did the phrase that was chosen by the baseline system,  $t_B$ . Thus, we ignored differences in the two translations that might have been due to the secondary effects of VSM adaptation (such as a different tar-

		no-lin-adap	lin-lm	lin-tm	lin-lm+lin-tm
Chinese	no vsm	31.7	32.1	32.7	33.1
	+vsm, joint	33.0**	33.1**	33.0	33.3
	+vsm, 3 feat.	33.1**	33.2**	33.1	<b>33.4</b>
Arabic	no vsm	46.8	47.0	47.5	47.7
	+vsm, joint	48.4**	48.7**	48.6**	48.8**
	+vsm, 3 feat.	48.6**	48.8**	48.7**	<b>48.9**</b>

Table 6: Results of combining VSM and linear mixture adaptation. “lin-lm” is linear language model adaptation, “lin-tm” is linear translation model adaptation. \* and \*\* denote significant gains over the row “no vsm” at  $p < 0.05$  and  $p < 0.01$  levels, respectively.

get phrase being preferred by the language model in the VSM-adapted system from the one preferred in the baseline system because of a Bhattacharyya-mediated change in the phrase preceding it).

An interesting pattern soon emerged: the VSM-adapted system seems to be better than the baseline at choosing among synonyms in a way that is appropriate to the genre or style of a text. For instance, where the text to be translated is from an informal genre such as weblog, the VSM-adapted system will often pick an informal word where the baseline picks a formal word with the same or similar meaning, and vice versa where the text to be translated is from a more formal genre. To our surprise, we saw few examples where the VSM-adapted system did a better job than the baseline of choosing between two words with different meaning, but we saw many examples where the VSM-adapted system did a better job than the baseline of choosing between two words that both have the same meaning according to considerations of style and genre.

Two examples are shown in Table 7. In the first example, the first two lines show that VSM finds that the Chinese-English phrase pair (殴打, assaulted) has a Bhattacharyya (BC) similarity of 0.556163 to the dev set, while the phrase pair (殴打, beat) has a BC similarity of 0.780787 to the dev. In this situation, the VSM-adapted system thus prefers “beat” to “assaulted” as a translation for 殴打. The next four lines show the source sentence (SRC), the reference (REF), the baseline output (BSL), and the output of the VSM-adapted system. Note that the result of VSM adaptation is that the rather formal word “assaulted” is replaced by its informal near-synonym “beat” in the translation of an informal weblog text.

“apprehend” might be preferable to “arrest” in a legal text. However, it looks as though the

VSM-adapted system has learned from the dev that among synonyms, those more characteristic of news stories than of legal texts should be chosen: it therefore picks “arrest” over its synonym “apprehend”.

What follows is a partial list of pairs of phrases (all single words) from our system’s outputs, where the baseline chose the first member of a pair and the VSM-adapted system chose the second member of the pair to translate the same Chinese phrase into English (because the second word yields a better BC score for the dev set we used). It will be seen that nearly all of the pairs involve synonyms or near-synonyms rather than words with radically different senses (one exception below is “center” vs “heart”). Instead, the differences between the two words tend to be related to genre or style: gunmen-mobsters, champion-star, updated-latest, caricatures-cartoons, spill-leakage, hiv-aids, inkling-clues, behaviour-actions, deceit-trick, brazen-shameless, aristocratic-noble, circumvent-avoid, attack-criticized, descent-born, hasten-quickly, precipice-cliff, center-heart, blessing-approval, imminent-approaching, stormed-rushed, etc.

## 4 Conclusions and future work

This paper proposed a new approach to domain adaptation in statistical machine translation, based on vector space models (VSMs). This approach measures the similarity between a vector representing a particular phrase pair in the phrase table and a vector representing the dev set, yielding a feature associated with that phrase pair that will be used by the decoder. The approach is simple, easy to implement, and computationally cheap. For the two language pairs we looked at, it provided a large performance improvement over a non-adaptive baseline, and also compared

1	phrase pairs	殴打 ↔ assaulted (0.556163) 殴打 ↔ beat (0.780787)
	SRC	...那些殴打村民的地皮流氓...
	REF	... those local ruffians and hooligans who beat up villagers ...
	BSL	... those who assaulted the villagers land hooligans ...
	VSM	... hooligans who beat the villagers ...
2	phrase pairs	缉拿 ↔ apprehend (0.286533) 缉拿 ↔ arrest (0.603342)
	SRC	... 缉拿凶手并且将之绳之以法。
	REF	... catch the killers and bring them to justice .
	BSL	... apprehend the perpetrators and bring them to justice .
	VSM	... arrest the perpetrators and bring them to justice .

Table 7: Examples show that VSM chooses translations according to considerations of style and genre.

favourably with linear mixture adaptation techniques.

Furthermore, VSM adaptation can be exploited in a number of different ways, which we have only begun to explore. In our experiments, we based the vector space on subcorpora defined by the nature of the training data. This was done purely out of convenience: there are many, many ways to define a vector space in this situation. An obvious and appealing one, which we intend to try in future, is a vector space based on a bag-of-words topic model. A feature derived from this topic-related vector space might complement some features derived from the subcorpora which we explored in the experiments above, and which seem to exploit information related to genre and style.

## References

- Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *EMNLP 2011*.
- Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the 4th Workshop on Statistical Machine Translation*, Athens, March. WMT.
- A. Bhattacharyya. 1943. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109.
- Sung-Hyuk Cha. 2007. Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4):300–307.
- Boxing Chen, Min Zhang, Aiti Aw, and Haizhou Li. 2008. Exploiting n-best hypotheses for smt self-enhancement. In *ACL 2008*.
- Boxing Chen, Roland Kuhn, George Foster, and Howard Johnson. 2011. Unpacking and transforming feature functions: New ways to smooth phrase tables. In *MT Summit 2011*.
- Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *NAACL 2012*.
- George Foster and Roland Kuhn. 2007. Mixture-model adaptation for SMT. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, Prague, June. WMT.
- George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Boston.
- Michel Galley and C. D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *EMNLP 2008*, pages 848–856, Hawaii, October.
- Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the 10th EAMT Conference*, Budapest, May.
- Donald Hindle. 1990. Noun classification from predicate.argument structures. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 268–275, Pittsburgh, PA, June. ACL.
- Fei Huang and Bing Xiang. 2010. Feature-rich discriminative phrase rescoring for SMT. In *COLING 2010*.
- Jun’ichi Kazama, Stijn De Saeger, Kow Kuroda, Masaki Murata, and Kentaro Torisawa. 2010. A

- bayesian method for robust estimation of distributional similarities. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 247–256, Uppsala, Sweden, July. ACL.
- Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Prague, Czech Republic, June. Association for Computational Linguistics.
- P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Demonstration Session*.
- P. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Barcelona, Spain.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING/ACL-98*, pages 768–774, Montreal, Quebec, Canada.
- Yajuan Lü, Jin Huang, and Qun Liu. 2007. Improving Statistical Machine Translation Performance by Training Data Selection and Optimization. In *Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Prague, Czech Republic.
- K. Lund and C. Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods Instruments and Computers*, 28(2):203–208.
- Spyros Matsoukas, Antti-Veikko I. Rosti, and Bing Zhang. 2009. Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Singapore.
- Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *ACL 2010*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, July. ACL.
- Aaron B. Phillips and Ralf D. Brown. 2011. Training machine translation with a second-order taylor approximation of weighted translation instances. In *MT Summit 2011*.
- Holger Schwenk. 2008. Investigations on large-scale lightly-supervised training for statistical machine translation. In *IWSLT 2008*.
- Rico Sennrich. 2012. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *EACL 2012*.
- Peter Turney. 2001. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *Twelfth European Conference on Machine Learning*, page 491–502, Berlin, Germany.
- Nicola Ueffing, Gholamreza Haffari, and Anoop Sarkar. 2007. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic, June. ACL.
- Bing Zhao, Matthias Eck, and Stephan Vogel. 2004. Language model adaptation for statistical machine translation with structured query models. In *Proceedings of the International Conference on Computational Linguistics (COLING) 2004*, Geneva, August.

# From Natural Language Specifications to Program Input Parsers

Tao Lei, Fan Long, Regina Barzilay, and Martin Rinard

Computer Science and Artificial Intelligence Laboratory  
Massachusetts Institute of Technology  
{taolei, fanl, regina, rinard}@csail.mit.edu

## Abstract

We present a method for automatically generating input parsers from English specifications of input file formats. We use a Bayesian generative model to capture relevant natural language phenomena and translate the English specification into a specification tree, which is then translated into a C++ input parser. We model the problem as a joint dependency parsing and semantic role labeling task. Our method is based on two sources of information: (1) the correlation between the text and the specification tree and (2) noisy supervision as determined by the success of the generated C++ parser in reading input examples. Our results show that our approach achieves 80.0% F-Score accuracy compared to an F-Score of 66.7% produced by a state-of-the-art semantic parser on a dataset of input format specifications from the ACM International Collegiate Programming Contest (which were written in English for humans with no intention of providing support for automated processing).<sup>1</sup>

## 1 Introduction

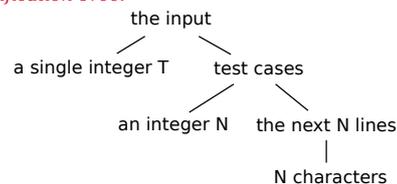
The general problem of translating natural language specifications into executable code has been around since the field of computer science was founded. Early attempts to solve this problem produced what were essentially verbose, clumsy, and ultimately unsuccessful versions of standard formal programming languages. In recent years

<sup>1</sup>The code, data, and experimental setup for this research are available at <http://groups.csail.mit.edu/rbg/code/nl2p>

### (a) Text Specification:

The input contains a single integer  $T$  that indicates the number of test cases. Then follow the  $T$  cases. Each test case begins with a line contains an integer  $N$ , representing the size of wall. The next  $N$  lines represent the original wall. Each line contains  $N$  characters. The  $j$ -th character of the  $i$ -th line figures out the color ...

### (b) Specification Tree:



### (c) Two Program Input Examples:

1	2
10	1
YYWYWWWWW	Y
YWWWYWWWWW	5
YWYWWWWW	YWYWW
...	...
WWWWWWWW	WWYY

Figure 1: An example of (a) one natural language specification describing program input data; (b) the corresponding specification tree representing the program input structure; and (c) two input examples

however, researchers have had success addressing specific aspects of this problem. Recent advances in this area include the successful translation of natural language commands into database queries (Wong and Mooney, 2007; Zettlemoyer and Collins, 2009; Poon and Domingos, 2009; Liang et al., 2011) and the successful mapping of natural language instructions into Windows command sequences (Branavan et al., 2009; Branavan et al., 2010).

In this paper we explore a different aspect of this general problem: the translation of natural language input specifications into executable code that correctly parses the input data and generates

data structures for holding the data. The need to automate this task arises because input format specifications are almost always described in natural languages, with these specifications then manually translated by a programmer into the code for reading the program inputs. Our method highlights potential to automate this translation, thereby eliminating the manual software development overhead.

Consider the text specification in Figure 1a. If the desired parser is implemented in C++, it should create a C++ class whose instance objects hold the different fields of the input. For example, one of the fields of this class is an integer, i.e., “a single integer T” identified in the text specification in Figure 1a. Instead of directly generating code from the text specification, we first translate the specification into a *specification tree* (see Figure 1b), then map this tree into parser code (see Figure 2). We focus on the translation from the text specification to the specification tree.<sup>2</sup>

We assume that each text specification is accompanied by a set of input examples that the desired input parser is required to successfully read. In standard software development contexts, such input examples are usually available and are used to test the correctness of the input parser. Note that this source of supervision is noisy — the generated parser may still be incorrect even when it successfully reads all of the input examples. Specifically, the parser may interpret the input examples differently from the text specification. For example, the program input in Figure 1c can be interpreted simply as a list of strings. The parser may also fail to parse some correctly formatted input files not in the set of input examples. Therefore, our goal is to design a technique that can effectively learn from this weak supervision.

We model our problem as a joint dependency parsing and role labeling task, assuming a Bayesian generative process. The distribution over the space of specification trees is informed by two sources of information: (1) the correlation between the text and the corresponding specification tree and (2) the success of the generated parser in reading input examples. Our method uses a joint probability distribution to take both of these sources of information into account, and uses a sampling framework for the inference of specifi-

<sup>2</sup>During the second step of the process, the specification tree is deterministically translated into code.

```

1  struct TestCaseType {
2      int N;
3      vector<NLinesType*> lstLines;
4      InputType* pParentLink;
5  }
6
7  struct InputType {
8      int T;
9      vector<TestCaseType*> lstTestCase;
10 }
11
12 TestCaseType* ReadTestCase(FILE * pStream,
13                             InputType* pParentLink) {
14     TestCaseType* pTestCase
15         = new TestCaseType;
16     pTestCase->pParentLink = pParentLink;
17
18     ...
19
20     return pTestCase;
21 }
22
23 InputType* ReadInput(FILE * pStream) {
24     InputType* plnput = new InputType;
25
26     plnput->T = ReadInteger(pStream);
27     for (int i = 0; i < plnput->T; ++i) {
28         TestCaseType* pTestCase
29             = new TestCaseType;
30         pTestCase = ReadTestCase (pStream,
31                                 plnput);
32         plnput->lstTestCase.push_back (pTestCase);
33     }
34
35     return plnput;
36 }

```

Figure 2: Input parser code for reading input files specified in Figure 1.

cation trees given text specifications. A specification tree is rejected in the sampling framework if the corresponding code fails to successfully read all of the input examples. The sampling framework also rejects the tree if the text/specification tree pair has low probability.

We evaluate our method on a dataset of input specifications from ACM International Collegiate Programming Contests, along with the corresponding input examples. These specifications were written for human programmers with no intention of providing support for automated processing. However, when trained using the noisy supervision, our method achieves substantially more accurate translations than a state-of-the-art semantic parser (Clarke et al., 2010) (specifically, 80.0% in F-Score compared to an F-Score of 66.7%). The strength of our model in the face of such weak supervision is also highlighted by the fact that it retains an F-Score of 77% even when only one input example is provided for each input

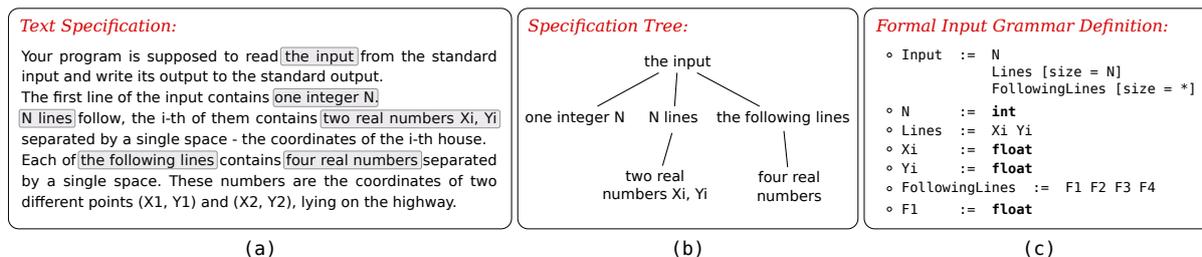


Figure 3: An example of generating input parser code from text: (a) a natural language input specification; (b) a specification tree representing the input format structure (we omit the background phrases in this tree in order to give a clear view of the input format structure); and (c) formal definition of the input format constructed from the specification tree, represented as a context-free grammar in Backus-Naur Form with additional size constraints.

specification.

## 2 Related Work

### Learning Meaning Representation from Text

Mapping sentences into structural meaning representations is an active and extensively studied task in NLP. Examples of meaning representations considered in prior research include logical forms based on database query (Tang and Mooney, 2000; Zettlemoyer and Collins, 2005; Kate and Mooney, 2007; Wong and Mooney, 2007; Poon and Domingos, 2009; Liang et al., 2011; Goldwasser et al., 2011), semantic frames (Das et al., 2010; Das and Smith, 2011) and database records (Chen and Mooney, 2008; Liang et al., 2009).

**Learning Semantics from Feedback** Our approach is related to recent research on learning from indirect supervision. Examples include leveraging feedback available via responses from a virtual world (Branavan et al., 2009) or from executing predicted database queries (Chang et al., 2010; Clarke et al., 2010). While Branavan et al. (2009) formalize the task as a sequence of decisions and learns from local rewards in a Reinforcement Learning framework, our model learns to predict the whole structure at a time. Another difference is the way our model incorporates the noisy feedback. While previous approaches rely on the feedback to train a *discriminative* prediction model, our approach models a *generative process* to guide structure predictions when the feedback is noisy or unavailable.

**NLP in Software Engineering** Researchers have recently developed a number of approaches that apply natural language processing techniques to software engineering problems. Examples include analyzing API documents to infer API li-

brary specifications (Zhong et al., 2009; Pandita et al., 2012) and analyzing code comments to detect concurrency bugs (Tan et al., 2007; Tan et al., 2011). This research analyzes natural language in documentation or comments to better understand existing application programs. Our mechanism, in contrast, automatically generates parser programs from natural language input format descriptions.

## 3 Problem Formulation

The task of translating text specifications to input parsers consists of two steps, as shown in Figure 3. First, given a text specification describing an input format, we wish to infer a parse tree (which we call a *specification tree*) implied by the text. Second, we convert each specification tree into formal grammar of the input format (represented in Backus-Naur Form) and then generate code that reads the input into data structures. In this paper, we focus on the NLP techniques used in the first step, i.e., learning to infer the specification trees from text. The second step is achieved using a deterministic rule-based tool.<sup>3</sup>

As input, we are given a set of text specifications  $w = \{w^1, \dots, w^N\}$ , where each  $w^i$  is a text specification represented as a sequence of noun phrases  $\{w_k^i\}$ . We use UIUC shallow parser to preprocess each text specification into a sequence of the noun phrases.<sup>4</sup> In addition, we are given a set of input examples for each  $w^i$ . We use these examples to test the generated input parsers to re-

<sup>3</sup>Specifically, the specification tree is first translated into the grammar using a set of rules and seed words that identifies basic data types such as `int`. Our implementation then generates a top-down parser since the generated grammar is simple. In general, standard techniques such as Bison and Yacc (Johnson, 1979) can generate bottom-up parsers given such grammar.

<sup>4</sup><http://cogcomp.cs.illinois.edu/demo/shallowparse/?id=7>

ject incorrect predictions made by our probabilistic model.

We formalize the learning problem as a dependency parsing and role labeling problem. Our model predicts specification trees  $\mathbf{t} = \{t^1, \dots, t^N\}$  for the text specifications, where each specification tree  $t^i$  is a dependency tree over noun phrases  $\{w_k^i\}$ . In general many program input formats are nested tree structures, in which the tree root denotes the entire chunk of program input data and each chunk (tree node) can be further divided into sub-chunks or primitive fields that appear in the program input (see Figure 3). Therefore our objective is to predict a dependency tree that correctly represents the structure of the program input.

In addition, the role labeling problem is to assign a tag  $z_k^i$  to each noun phrase  $w_k^i$  in a specification tree, indicating whether the phrase is a *key phrase* or a *background phrase*. Key phrases are named entities that identify input fields or input chunks appear in the program input data, such as “the input” or “the following lines” in Figure 3b. In contrast, background phrases do not define input fields or chunks. These phrases are used to organize the document (e.g., “your program”) or to refer to key phrases described before (e.g., “each line”).

## 4 Model

We use two kinds of information to bias our model: (1) the quality of the generated code as measured by its ability to read the given input examples and (2) the features over the observed text  $w^i$  and the hidden specification tree  $t^i$  (this is standard in traditional parsing problems). We combine these two kinds of information into a Bayesian generative model in which the code quality of the specification tree is captured by the prior probability  $P(\mathbf{t})$  and the feature observations are encoded in the likelihood probability  $P(\mathbf{w}|\mathbf{t})$ . The inference jointly optimizes these two factors:

$$P(\mathbf{t}|\mathbf{w}) \propto P(\mathbf{t}) \cdot P(\mathbf{w}|\mathbf{t}).$$

**Modeling the Generative Process.** We assume the generative model operates by first generating the model parameters from a set of Dirichlet distributions. The model then generates text specification trees. Finally, it generates natural language feature observations conditioned on the hidden specification trees.

The generative process is described formally as follows:

- **Generating Model Parameters:** For every pair of feature type  $f$  and phrase tag  $z$ , draw a multinomial distribution parameter  $\theta_f^z$  from a Dirichlet prior  $P(\theta_f^z)$ . The multinomial parameters provide the probabilities of observing different feature values in the text.
- **Generating Specification Tree:** For each text specification, draw a specification tree  $t$  from all possible trees over the sequence of noun phrases in this specification. We denote the probability of choosing a particular specification tree  $t$  as  $P(t)$ .

Intuitively, this distribution should assign high probability to good specification trees that can produce C++ code that reads all input examples without errors, we therefore define  $P(t)$  as follows:<sup>5</sup>

$$P(t) = \frac{1}{Z} \cdot \begin{cases} 1 & \text{the input parser of tree } t \\ & \text{reads all input examples} \\ & \text{without error} \\ \epsilon & \text{otherwise} \end{cases}$$

where  $Z$  is a normalization factor and  $\epsilon$  is empirically set to  $10^{-6}$ . In other words,  $P(\cdot)$  treats all specification trees that pass the input example test as equally probable candidates and inhibits the model from generating trees which fail the test. Note that we do not know this distribution a priori until the specification trees are evaluated by testing the corresponding C++ code. Because it is intractable to test all possible trees and all possible generated code for a text specification, we never explicitly compute the normalization factor  $1/Z$  of this distribution. We therefore use sampling methods to tackle this problem during inference.

- **Generating Features:** The final step generates lexical and contextual features for each tree. For each phrase  $w_k$  associated with tag  $z_k$ , let  $w_p$  be its parent phrase in the tree and  $w_s$  be the non-background sibling phrase to its left in the tree. The model generates the corresponding set of features  $\phi(w_p, w_s, w_k)$  for each text phrase tuple  $(w_p, w_s, w_k)$ , with

<sup>5</sup>When input examples are not available,  $P(t)$  is just uniform distribution.

probability  $P(\phi(w_p, w_s, w_k))$ . We assume that each feature  $f_j$  is generated independently:

$$\begin{aligned} P(w|t) &= P(\phi(w_p, w_s, w_k)) \\ &= \prod_{f_j \in \phi(w_p, w_s, w_k)} \theta_{f_j}^{z_k} \end{aligned}$$

where  $\theta_{f_j}^{z_k}$  is the  $j$ -th component in the multinomial distribution  $\theta_f^{z_k}$  denoting the probability of observing a feature  $f_j$  associated with noun phrase  $w_k$  labeled with tag  $z_k$ . We define a range of features that capture the correspondence between the input format and its description in natural language. For example, at the unigram level we aim to capture that noun phrases containing specific words such as “cases” and “lines” may be key phrases (correspond to data chunks appear in the input), and that verbs such as “contain” may indicate that the next noun phrase is a key phrase.

The full joint probability of a set  $\mathbf{w}$  of  $N$  specifications and hidden text specification trees  $\mathbf{t}$  is defined as:

$$\begin{aligned} P(\theta, \mathbf{t}, \mathbf{w}) &= P(\theta) \prod_{i=1}^N P(t^i) P(w^i | t^i, \theta) \\ &= P(\theta) \prod_{i=1}^N P(t^i) \prod_k P(\phi(w_p^i, w_s^i, w_k^i)). \end{aligned}$$

**Learning the Model** During inference, we want to estimate the hidden specification trees  $\mathbf{t}$  given the observed natural language specifications  $\mathbf{w}$ , after integrating the model parameters out, i.e.

$$\mathbf{t} \sim P(\mathbf{t} | \mathbf{w}) = \int_{\theta} P(\mathbf{t}, \theta | \mathbf{w}) d\theta.$$

We use Gibbs sampling to sample variables  $\mathbf{t}$  from this distribution. In general, the Gibbs sampling algorithm randomly initializes the variables and then iteratively solves one subproblem at a time. The subproblem is to sample only one variable conditioned on the current values of all other variables. In our case, we sample one hidden specification tree  $t^i$  while holding all other trees  $\mathbf{t}^{-i}$  fixed:

$$t^i \sim P(t^i | \mathbf{w}, \mathbf{t}^{-i}) \quad (1)$$

where  $\mathbf{t}^{-i} = (t^1, \dots, t^{i-1}, t^{i+1}, \dots, t^N)$ .

However directly solving the subproblem (1) in our case is still hard, we therefore use a Metropolis-Hastings sampler that is similarly applied in traditional sentence parsing problems. Specifically, the Hastings sampler approximates (1) by first drawing a new  $t^{i'}$  from a tractable proposal distribution  $Q$  instead of  $P(t^i | \mathbf{w}, \mathbf{t}^{-i})$ . We choose  $Q$  to be:

$$Q(t^{i'} | \theta', w^i) \propto P(w^i | t^{i'}, \theta'). \quad (2)$$

Then the probability of accepting the new sample is determined using the typical Metropolis Hastings process. Specifically,  $t^{i'}$  will be accepted to replace the last  $t^i$  with probability:

$$\begin{aligned} R(t^i, t^{i'}) &= \min \left\{ 1, \frac{P(t^{i'} | \mathbf{w}, \mathbf{t}^{-i}) Q(t^i | \theta', w^i)}{P(t^i | \mathbf{w}, \mathbf{t}^{-i}) Q(t^{i'} | \theta', w^i)} \right\} \\ &= \min \left\{ 1, \frac{P(t^{i'}, \mathbf{t}^{-i}, \mathbf{w}) P(w^i | t^i, \theta')}{P(t^i, \mathbf{t}^{-i}, \mathbf{w}) P(w^i | t^{i'}, \theta')} \right\}, \end{aligned}$$

in which the normalization factors  $1/Z$  are cancelled out. We choose  $\theta'$  to be the parameter expectation based on the current observations, i.e.  $\theta' = E[\theta | \mathbf{w}, \mathbf{t}^{-i}]$ , so that the proposal distribution is close to the true distribution. This sampling algorithm with a changing proposal distribution has been shown to work well in practice (Johnson and Griffiths, 2007; Cohn et al., 2010; Naseem and Barzilay, 2011). The algorithm pseudo code is shown in Algorithm 1.

To sample from the proposal distribution (2) efficiently, we implement a dynamic programming algorithm which calculates marginal probabilities of all subtrees. The algorithm works similarly to the inside algorithm (Baker, 1979), except that we do not assume the tree is binary. We therefore perform one additional dynamic programming step that sums over all possible segmentations of each span. Once the algorithm obtains the marginal probabilities of all subtrees, a specification tree can be drawn recursively in a top-down manner.

Calculating  $P(\mathbf{t}, \mathbf{w})$  in  $R(t, t')$  requires integrating the parameters  $\theta$  out. This has a closed form due to the Dirichlet-multinomial conjugacy:

$$\begin{aligned} P(\mathbf{t}, \mathbf{w}) &= P(\mathbf{t}) \cdot \int_{\theta} P(\mathbf{w} | \mathbf{t}, \theta) P(\theta) d\theta \\ &\propto P(\mathbf{t}) \cdot \prod \text{Beta}(\text{count}(f) + \alpha). \end{aligned}$$

Here  $\alpha$  are the Dirichlet hyper parameters and  $\text{count}(f)$  are the feature counts observed in data  $(\mathbf{t}, \mathbf{w})$ . The closed form is a product of the Beta functions of each feature type.

Feature Type	Description	Feature Value
Word	each word in noun phrase $w_k$	lines, VAR
Verb	verbs in noun phrase $w_k$ and the verb phrase before $w_k$	contains
Distance	sentence distance between $w_k$ and its parent phrase $w_p$	1
Coreference	$w_k$ share duplicate nouns or variable names with $w_p$ or $w_s$	True

Table 1: Example of feature types and values. To deal with sparsity, we map variable names such as “N” and “X” into a category word “VAR” in word features.

**Input:** Set of text specification documents  
 $\mathbf{w} = \{w^1, \dots, w^N\}$ ,  
Number of iterations  $T$

- 1 Randomly initialize specification trees  
 $\mathbf{t} = \{t^1, \dots, t^N\}$
- 2 **for**  $iter = 1 \dots T$  **do**
- 3   *Sample tree  $t^i$  for  $i$ -th document:*
- 4   **for**  $i = 1 \dots N$  **do**
- 5     *Estimate model parameters:*
- 6      $\theta' = E[\theta' | \mathbf{w}, \mathbf{t}^{-i}]$
- 7     *Sample a new specification tree from distribution  $Q$ :*
- 8      $t' \sim Q(t' | \theta', w^i)$
- 9     *Generate and test code, and return feedback:*
- 10      $f' = \text{CodeGenerator}(w^i, t')$
- 11     *Calculate accept probability  $r$ :*
- 12      $r = R(t^i, t')$
- 13     *Accept the new tree with probability  $r$ :*
- 14     With probability  $r$ :  $t^i = t'$
- 15   **end**
- 16 **end**
- 17 *Produce final structures:*
- 18 **return**  $\{t^i \text{ if } t^i \text{ gets positive feedback}\}$

Algorithm 1: The sampling framework for learning the model.

**Model Implementation:** We define several types of features to capture the correlation between the hidden structure and its expression in natural language. For example, verb features are introduced because certain preceding verbs such as “contains” and “consists” are good indicators of key phrases. There are 991 unique features in total in our experiments. Examples of features appear in Table 1.

We use a small set of 8 seed words to bias the search space. Specifically, we require each leaf key phrase to contain at least one seed word that identifies the C++ primitive data type (such as “integer”, “float”, “byte” and “string”).

We also encourage a phrase containing the word “input” to be the root of the tree (for example, “the input file”) and each coreference phrase to be a

Total # of words	7330
Total # of noun phrases	1829
Vocabulary size	781
Avg. # of words per sentence	17.29
Avg. # of noun phrase per document	17.26
Avg. # of possible trees per document	52K
Median # of possible trees per document	79
Min # of possible trees per document	1
Max # of possible trees per document	2M

Table 2: Statistics for 106 ICPC specifications.

background phrase (for example, “each test case” after mentioning “test cases”), by initially adding pseudo counts to Dirichlet priors.

## 5 Experimental Setup

**Datasets:** Our dataset consists of problem descriptions from ACM International Collegiate Programming Contests.<sup>6</sup> We collected 106 problems from ACM-ICPC training websites.<sup>7</sup> From each problem description, we extracted the portion that provides input specifications. Because the test input examples are not publicly available on the ACM-ICPC training websites, for each specification, we wrote simple programs to generate 100 random input examples.

Table 2 presents statistics for the text specification set. The data set consists of 424 sentences, where an average sentence contains 17.3 words. The data set contains 781 unique words. The length of each text specification varies from a single sentence to eight sentences. The difference between the average and median number of trees is large. This is because half of the specifications are relatively simple and have a small number of possible trees, while a few difficult specifications have over thousands of possible trees (as the number of trees grows exponentially when the text length increases).

**Evaluation Metrics:** We evaluate the model

<sup>6</sup>Official Website: <http://cm.baylor.edu/welcome.icpc>

<sup>7</sup>PKU Online Judge: <http://poj.org/>; UVA Online Judge: <http://uva.onlinejudge.org/>

performance in terms of its success in generating a formal grammar that correctly represents the input format (see Figure 3c). As a gold annotation, we construct formal grammars for all text specifications. Our results are generated by automatically comparing the machine-generated grammars with their golden counterparts. If the formal grammar is correct, then the generated C++ parser will correctly read the input file into corresponding C++ data structures.

We use Recall and Precision as evaluation measures:

$$\text{Recall} = \frac{\# \text{ correct structures}}{\# \text{ text specifications}}$$

$$\text{Precision} = \frac{\# \text{ correct structures}}{\# \text{ produced structures}}$$

where the produced structures are the positive structures returned by our framework whose corresponding code successfully reads all input examples (see Algorithm 1 line 18). Note the number of produced structures may be less than the number of text specifications, because structures that fail the input test are not returned.

**Baselines:** To evaluate the performance of our model, we compare against four baselines.

The *No Learning* baseline is a variant of our model that selects a specification tree without learning feature correspondence. It continues sampling a specification tree for each text specification until it finds one which successfully reads all of the input examples.

The second baseline *Aggressive* is a state-of-the-art semantic parsing framework (Clarke et al., 2010).<sup>8</sup> The framework repeatedly predicts hidden structures (specification trees in our case) using a structure learner, and trains the structure learner based on the execution feedback of its predictions. Specifically, at each iteration the structure learner predicts the most plausible specification tree for each text document:

$$t^i = \operatorname{argmax}_t f(w^i, t).$$

Depending on whether the corresponding code reads all input examples successfully or not, the  $(w^i, t^i)$  pairs are added as an positive or negative sample to populate a training set. After each iteration the structure learner is re-trained with the training samples to improve the prediction accuracy. In our experiment, we follow (Clarke et al.,

<sup>8</sup>We take the name *Aggressive* from this paper.

Model	Recall	Precision	F-Score
No Learning	52.0	57.2	54.5
Aggressive	63.2	70.5	66.7
<b>Full Model</b>	<b>72.5</b>	<b>89.3</b>	<b>80.0</b>
Full Model (Oracle)	72.5	100.0	84.1
Aggressive (Oracle)	80.2	100.0	89.0

Table 3: Average % Recall and % Precision of our model and all baselines over 20 independent runs.

2010) and choose a structural Support Vector Machine SVM<sup>struct</sup><sup>9</sup> as the structure learner.

The remaining baselines provide an upper bound on the performance of our model. The baseline *Full Model (Oracle)* is the same as our full model except that the feedback comes from an oracle which tells whether the specification tree is correct or not. We use this oracle information in the prior  $P(t)$  same as we use the noisy feedback. Similarly the baseline *Aggressive (Oracle)* is the *Aggressive* baseline with access to the oracle.

**Experimental Details:** Because no human annotation is required for learning, we train our model and all baselines on all 106 ICPC text specifications (similar to unsupervised learning). We report results averaged over 20 independent runs. For each of these runs, the model and all baselines run 100 iterations. For baseline *Aggressive*, in each iteration the SVM structure learner predicts one tree with the highest score for each text specification. If two different specification trees of the same text specification get positive feedback, we take the one generated in later iteration for evaluation.

## 6 Experimental Results

**Comparison with Baselines** Table 3 presents the performance of various models in predicting correct specification trees. As can be seen, our model achieves an F-Score of 80%. Our model therefore significantly outperforms the *No Learning* baseline (by more than 25%). Note that the *No Learning* baseline achieves a low Precision of 57.2%. This low precision reflects the noisiness of the weak supervision - nearly one half of the parsers produced by *No Learning* are actually incorrect even though they read all of the input examples without error. This comparison shows the importance of capturing correlations between the specification trees and their text descriptions.

<sup>9</sup>[www.cs.cornell.edu/people/tj/svm\\_light/svm\\_struct.html](http://www.cs.cornell.edu/people/tj/svm_light/svm_struct.html)

- (a) The input contains several testcases. Each is specified by two strings S, T of alphanumeric ASCII characters
- (b) The next N lines of the input file contain the Cartesian coordinates of watchtowers, one pair of coordinates per line.

Figure 4: Examples of dependencies and key phrases predicted by our model. Green marks correct key phrases and dependencies and red marks incorrect ones. The missing key phrases are marked in gray.

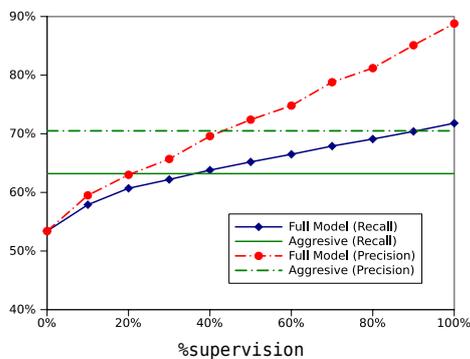


Figure 5: Precision and Recall of our model by varying the percentage of weak supervision. The green lines are the performance of *Aggressive* baseline trained with full weak supervision.

Because our model learns correlations via feature representations, it produces substantially more accurate translations.

While both the *Full Model* and *Aggressive* baseline use the same source of feedback, they capitalize on it in a different way. The baseline uses the noisy feedback to train features capturing the correlation between trees and text. Our model, in contrast, combines these two sources of information in a complementary fashion. This combination allows our model to filter false positive feedback and produce 13% more correct translations than the *Aggressive* baseline.

**Clean versus Noisy Supervision** To assess the impact of noise on model accuracy, we compare the *Full Model* against the *Full Model (Oracle)*. The two versions achieve very close performance (80% v.s 84% in F-Score), even though *Full Model* is trained with noisy feedback. This demonstrates the strength of our model in learning from such weak supervision. Interestingly, *Aggressive (Oracle)* outperforms our oracle model by a 5% margin. This result shows that when the supervision is reliable, the generative assumption limits our model’s ability to gain the same performance improvement as discriminative models.

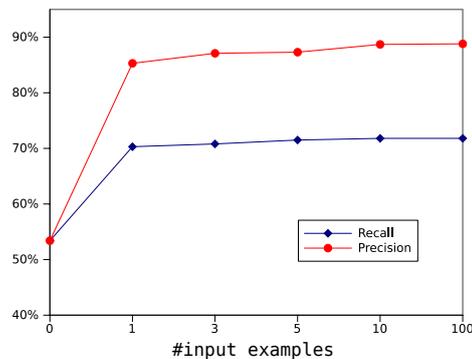


Figure 6: Precision and Recall of our model by varying the number of available input examples per text specification.

**Impact of Input Examples** Our model can also be trained in a fully unsupervised or a semi-supervised fashion. In real cases, it may not be possible to obtain input examples for all text specifications. We evaluate such cases by varying the amount of supervision, i.e. how many text specifications are paired with input examples. In each run, we randomly select text specifications and only these selected specifications have access to input examples. Figure 5 gives the performance of our model with 0% supervision (totally unsupervised) to 100% supervision (our full model). With much less supervision, our model is still able to achieve performance comparable with the *Aggressive* baseline.

We also evaluate how the number of provided input examples influences the performance of the model. Figure 6 indicates that the performance is largely insensitive to the number of input examples — once the model is given even one input example, its performance is close to the best performance it obtains with 100 input examples. We attribute this phenomenon to the fact that if the generated code is incorrect, it is unlikely to successfully parse any input.

**Case Study** Finally, we consider some text specifications that our model does not correctly trans-

late. In Figure 4a, the program input is interpreted as a list of character strings, while the correct interpretation is that the input is a list of string pairs. Note that both interpretations produce C++ input parsers that successfully read all of the input examples. One possible way to resolve this problem is to add other features such as syntactic dependencies between words to capture more language phenomena. In Figure 4b, the missing key phrase is not identified because our model is not able to ground the meaning of “pair of coordinates” to two integers. Possible future extensions to our model include using lexicon learning methods for mapping words to C++ primitive types for example “coordinates” to `<int, int>`.

## 7 Conclusion

It is standard practice to write English language specifications for input formats. Programmers read the specifications, then develop source code that parses inputs in the format. Known disadvantages of this approach include development cost, parsers that contain errors, specification misunderstandings, and specifications that become out of date as the implementation evolves.

Our results show that taking both the correlation between the text and the specification tree and the success of the generated C++ parser in reading input examples into account enables our method to correctly generate C++ parsers for 72.5% of our natural language specifications.

## 8 Acknowledgements

The authors acknowledge the support of Battelle Memorial Institute (PO #300662) and the NSF (Grant IIS-0835652). Thanks to Mirella Lapata, members of the MIT NLP group and the ACL reviewers for their suggestions and comments. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors, and do not necessarily reflect the views of the funding organizations.

## References

James K. Baker. 1979. Trainable grammars for speech recognition. In DH Klatt and JJ Wolf, editors, *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550.

S. R. K. Branavan, Harr Chen, Luke S. Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning

for mapping instructions to actions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

- S.R.K Branavan, Luke Zettlemoyer, and Regina Barzilay. 2010. Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of ACL*, pages 1268–1277.
- Mingwei Chang, Vivek Srikumar, Dan Goldwasser, and Dan Roth. 2010. Structured output learning with indirect supervision. In *Proceedings of the 27th International Conference on Machine Learning*.
- David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of 25th International Conference on Machine Learning (ICML-2008)*.
- James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world’s response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*.
- Trevor Cohn, Phil Blunsom, and Sharon Goldwater. 2010. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11.
- Dipanjan Das and Noah A. Smith. 2011. Semi-supervised frame-semantic parsing for unknown predicates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1435–1444.
- Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 948–956.
- Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT ’11*.
- Mark Johnson and Thomas L. Griffiths. 2007. Bayesian inference for pcfgs via markov chain monte carlo. In *Proceedings of the North American Conference on Computational Linguistics (NAACL ’07)*.
- Stephen C. Johnson. 1979. Yacc: Yet another compiler-compiler. *Unix Programmer’s Manual*, vol 2b.
- Rohit J. Kate and Raymond J. Mooney. 2007. Learning language semantics from ambiguous supervision. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 1, AAAI’07*.

- P. Liang, M. I. Jordan, and D. Klein. 2009. Learning semantic correspondences with less supervision. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*.
- P. Liang, M. I. Jordan, and D. Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- Tahira Naseem and Regina Barzilay. 2011. Using semantic cues to learn syntax. In *Proceedings of the 25th National Conference on Artificial Intelligence (AAAI)*.
- Rahul Pandita, Xusheng Xiao, Hao Zhong, Tao Xie, Stephen Oney, and Amit Paradkar. 2012. Inferring method specifications from natural language api descriptions. In *Proceedings of the 2012 International Conference on Software Engineering, ICSE 2012*, pages 815–825, Piscataway, NJ, USA. IEEE Press.
- Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1, EMNLP '09*.
- Lin Tan, Ding Yuan, Gopal Krishna, and Yuanyuan Zhou. 2007. /\* iComment: Bugs or bad comments? \*/. In *Proceedings of the 21st ACM Symposium on Operating Systems Principles (SOSP07)*, October.
- Lin Tan, Yuanyuan Zhou, and Yoann Padioleau. 2011. aComment: Mining annotations from comments and code to detect interrupt-related concurrency bugs. In *Proceedings of the 33rd International Conference on Software Engineering (ICSE11)*, May.
- Lappoon R. Tang and Raymond J. Mooney. 2000. Automated construction of database interfaces: integrating statistical and relational learning for semantic parsing. In *Proceedings of the conference on Empirical Methods in Natural Language Processing, EMNLP '00*.
- Yuk Wah Wong and Raymond J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *ACL*.
- Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of UAI*, pages 658–666.
- Luke S. Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- Hao Zhong, Lu Zhang, Tao Xie, and Hong Mei. 2009. Inferring resource specifications from natural language api documentation. In *Proceedings of the 2009 IEEE/ACM International Conference on Automated Software Engineering, ASE '09*, pages 307–318, Washington, DC, USA. IEEE Computer Society.

# Entity Linking for Tweets

Xiaohua Liu<sup>†</sup>, Yitong Li<sup>‡</sup>, Haocheng Wu<sup>#</sup>, Ming Zhou<sup>†</sup>, Furu Wei<sup>†</sup>, Yi Lu<sup>§</sup>

<sup>†</sup>Microsoft Research Asia, Beijing, 100190, China

<sup>‡</sup>School of Electronic and Information Engineering  
Beihang University, Beijing, 100191, China

<sup>#</sup>University of Science and Technology of China  
No. 96, Jinzhai Road, Hefei, Anhui, China

<sup>§</sup>School of Computer Science and Technology  
Harbin Institute of Technology, Harbin, 150001, China

<sup>†</sup>{xiaoliu, mingzhou, fuwei}@microsoft.com

<sup>‡</sup>tong91222@126.com <sup>#</sup>v-haowu@microsoft.com <sup>§</sup>v-y@microsoft.com

## Abstract

We study the task of entity linking for tweets, which tries to associate each mention in a tweet with a knowledge base entry. Two main challenges of this task are the dearth of information in a single tweet and the rich entity mention variations. To address these challenges, we propose a collective inference method that simultaneously resolves a set of mentions. Particularly, our model integrates three kinds of similarities, i.e., mention-entry similarity, entry-entry similarity, and mention-mention similarity, to enrich the context for entity linking, and to address irregular mentions that are not covered by the entity-variation dictionary. We evaluate our method on a publicly available data set and demonstrate the effectiveness of our method.

## 1 Introduction

Twitter is a widely used social networking service. With millions of active users and hundreds of millions of new published tweets every day<sup>1</sup>, it has become a popular platform to capture and transmit the human experiences of the moment. Many tweet related researches are inspired, from named entity recognition (Liu et al., 2012), topic detection (Mathioudakis and Koudas, 2010), clustering (Rosa et al., 2010), to event extraction (Grinev et al., 2009).

In this work, we study the entity linking task for tweets, which maps each entity mention in a tweet to a unique entity, i.e., an entry ID of a knowledge base like Wikipedia. Entity

linking task is generally considered as a bridge between unstructured text and structured machine-readable knowledge base, and represents a critical role in machine reading program (Singh et al., 2011). Entity linking for tweets is particularly meaningful, considering that tweets are often hard to read owing to its informal written style and length limitation of 140 characters.

Current entity linking methods are built on top of a large scale knowledge base such as Wikipedia. A knowledge base consists of a set of entities, and each entity can have a variation list<sup>2</sup>. To decide which entity should be mapped, they may compute: 1) the similarity between the context of a mention, e.g., a text window around the mention, and the content of an entity, e.g., the entity page of Wikipedia (Mihalcea and Csomai, 2007; Han and Zhao, 2009); 2) the coherence among the mapped entities for a set of related mentions, e.g. multiple mentions in a document (Milne and Witten, 2008; Kulkarni et al., 2009; Han and Zhao, 2010; Han et al., 2011).

Tweets pose special challenges to entity linking. First, a tweet is often too concise and too noisy to provide enough information for similarity computing, owing to its short and grass root nature. Second, tweets have rich variations of named entities<sup>3</sup>, and many of them fall out of the scope of the existing dictionaries mined from Wikipedia (called OOV mentions hereafter). On

<sup>2</sup>Entity variation lists can be extracted from the entity resolution pages of Wikipedia. For example, the link “<http://en.wikipedia.org/wiki/Svm>” will lead us to a resolution page, where “Svm” are linked to entities like “Space vector modulation” and “Support vector machine”. As a result, “Svm” will be added into the variation lists of “Space vector modulation” and “Support vector machine”, respectively.

<sup>3</sup>According to Liu et al. (2012), on average a named entity has 3.3 different surface forms in tweets.

<sup>1</sup><http://siteanalytics.compete.com/twitter.com/>

the other hand, the huge redundancy in tweets offers opportunities. That means, an entity mention often occurs in many tweets, which allows us to aggregate all related tweets to compute mention-mention similarity and mention-entity similarity.

We propose a collective inference method that leverages tweet redundancy to address those two challenges. Given a set of mentions, our model tries to ensure that similar mentions are linked to similar entities while pursuing the high total similarity between matched mention-entity pairs. More specifically, we define local features, including context similarity and edit distance, to model the similarity between a mention and an entity. We adopt in-link based similarity (Milne and Witten, 2008), to measure the similarity between entities. Finally, we introduce a set of features to compute the similarity between mentions, including how similar the tweets containing the mentions are, whether they come from the tweets of the same account, and their edit distance. Notably, our model can resolve OOV mentions with the help of their similar mentions. For example, for the OOV mention “LukeBryanOnline”, our model can find similar mentions like “TheLukeBryan” and “LukeBryan”. Considering that most of its similar mentions are mapped to the American country singer “Luke Bryan”, our model tends to link “LukeBryanOnline” to the same entity.

We evaluate our method on the public available data set shared by Meij et al. (2012)<sup>4</sup>. Experimental results show that our method outperforms two baselines, i.e., Wikify! (Mihalcea and Csomai, 2007) and system proposed by Meij et al. (2012). We also study the effectiveness of features related to each kind of similarity, and demonstrate the advantage of our method for OOV mention linkage.

We summarize our contributions as follows.

1. We introduce a novel collective inference method that integrates three kinds of similarities, i.e., mention-entity similarity, entity-entity similarity, and mention-mention similarity, to simultaneously map a set of tweet mentions to their proper entities.
2. We propose modeling the mention-mention similarity and demonstrate its effectiveness

<sup>4</sup><http://ilps.science.uva.nl/resources/wsdm2012-adding-semantic-to-microblog-posts/>

in entity linking for tweets, particularly for OOV mentions.

3. We evaluate our method on a public data set, and show our method compares favorably with the baselines.

Our paper is organized as follows. In the next section, we introduce related work. In Section 3, we give the formal definition of the task. In Section 4, we present our solution, including the framework, features related to different kinds of similarities, and the training and decoding procedures. We evaluate our method in Section 5. Finally in Section 6, we conclude with suggestions of future work.

## 2 Related Work

Existing entity linking work can roughly be divided into two categories. Methods of the first category resolve one mention at each time, and mainly consider the similarity between a mention-entity pair. In contrast, methods of the second category take a set of related mentions (e.g., mentions in the same document) as input, and figure out their corresponding entities simultaneously.

Examples of the first category include the first Web-scale entity linking system SemTag (Dill et al., 2003), Wikify! (Mihalcea and Csomai, 2007), and the recent work of Milne and Witten (2008). SemTag uses the TAP knowledge base<sup>5</sup>, and employs the cosine similarity with TF-IDF weighting scheme to compute the match degree between a mention and an entity, achieving an accuracy of around 82%. Wikify! identifies the important concepts in the text and automatically links these concepts to the corresponding Wikipedia pages. It introduces two approaches to define mention-entity similarity, i.e., the contextual overlap between the paragraph where the mention occurs and the corresponding Wikipedia pages, and a Naive Bayes classifier that predicts whether a mention should be linked to an entity. It achieves 80.69% F1 when two approaches are combined. Milne and Witten work on the same task of Wikify!, and also train a classifier. However, they cleverly use the

<sup>5</sup>TAB (<http://www.w3.org/2002/05/tap/>) is a shallow knowledge base that contains a broad range of lexical and taxonomic information about popular objects like music, movies, authors, sports, autos, health, etc.

links found within Wikipedia articles for training, exploiting the fact that for every link, a Wikipedian has manually selected the correct destination to represent the intended sense of the anchor. Their method achieves an F1 score of 75.0%.

Representative studies of the second category include the work of Kulkarni et al. (2009), Han et al. (2011), and Shen et al. (2012). One common feature of these studies is that they leverage the global coherence between entities. Kulkarni et al. (2009) propose a graphical model that explicitly models the combination of evidence from local mention-entity compatibility and global document-level topical coherence of the entities, and show that considering global coherence between entities significantly improves the performance. Han et al. (2011) introduce a graph-based representation, called Referent Graph, to model the global interdependence between different entity linking decisions, and jointly infer the referent entities of all name mentions in a document by exploiting the interdependence captured in Referent Graph. Shen et al. (2012) propose LIEGE, a framework to link the entities in web lists with the knowledge base, with the assumption that entities mentioned in a Web list tend to be a collection of entities of the same conceptual type.

Most work of entity linking focuses on web pages. Recently, Meij et al. (2012) study this task for tweets. They propose a machine learning based approach using n-gram features, concept features, and tweet features, to identify concepts semantically related to a tweet, and for every entity mention to generate links to its corresponding Wikipedia article. Their method belongs to the first category, in the sense that they only consider the similarity between mention (tweet) and entity (Wikipedia article).

Our method belongs to the second category. However, in contrast with existing collective approaches, our method works on tweets which are short and often noisy. Furthermore, our method is based on the “similar mention with similar entity” assumption, and explicitly models and integrates the mention similarity into the optimization framework. Compared with Meij et al. (2012), our method is collective, and integrates more features.

### 3 Task Definition

Given a sequence of mentions, denoted by  $\vec{M} = (m_1, m_2, \dots, m_n)$ , our task is to output a sequence of entities, denoted by  $\vec{E} = (e_1, e_2, \dots, e_n)$ , where  $e_i$  is the entity corresponding to  $m_i$ . Here, an entity refers to an item of a knowledge base. Following most existing work, we use Wikipedia as the knowledge base, and an entity is a definition page in Wikipedia; a mention denotes a sequence of tokens in a tweet that can be potentially linked to an entity.

Several notes should be made. First, we assume that mentions are given, e.g., identified by some named entity recognition system. Second, mentions may come from multiple tweets. Third, mentions with the same token sequence may refer to different entities, depending on mention context. Finally, we assume each entity  $e$  has a variation list<sup>6</sup>, and a unique ID through which all related information about that entity can be accessed.

Here is an example to illustrate the task. Given mentions “nbcnightlynews”, “Santiago”, “WH” and “Libya” from the following tweet “Chuck Todd: Prepping for @nbcnightlynews here in Santiago, reporting on WH handling of Libya situation.”, the expected output is “NBC Nightly News(194735)”, “Santiago Chile(51572)”, “White House(33057)” and “Libya(17633)”, where the numbers in the parentheses are the IDs of the corresponding entities.

### 4 Our Method

In this section, we first present the framework of our entity linking method. Then we introduce features related to different kinds of similarities, followed by a detailed discussion of the training and decoding procedures.

#### 4.1 Framework

Given the input mention sequence  $\vec{M} = (m_1, m_2, \dots, m_n)$ , our method outputs the entity sequence  $\vec{E}^* = (e_1^*, e_2^*, \dots, e_n^*)$  according to Formula 1:

<sup>6</sup>For example, the variation list of the entity “Obama” may contain “Barack Obama”, “Barack Hussein Obama II”, etc.

$$\vec{E}^* = \underset{\vec{E} \in C(\vec{M})}{\operatorname{argmax}} \lambda \sum_{i=1}^n \vec{w} \cdot \vec{f}(e_i, m_i) + (1 - \lambda) \sum_{i \neq j} r(e_i, e_j) s(m_i, m_j) \quad (1)$$

Where:

- $C(\vec{M})$  is the set of all possible entity sequences for the mention sequence  $\vec{M}$ ;
- $\vec{E}$  denotes an entity sequence instance, consisting of  $e_1, e_2, \dots, e_n$ ;
- $\vec{f}(e_i, m_i)$  is the feature vector that models the similarity between mention  $m_i$  and its linked entity  $e_i$ ;
- $\vec{w}$  is the feature weight vector related to  $\vec{f}$ , which is trained on the training data set;  $\vec{w} \cdot \vec{f}(e_i, m_i)$  is the similarity between mention  $m_i$  and entity  $e_i$ ;
- $r(e_i, e_j)$  is the function that returns the similarity between two entities  $e_i$  and  $e_j$ ;
- $s(m_i, m_j)$  is the function that returns the similarity between two mentions  $m_i$  and  $m_j$ ;
- $\lambda \in (0, 1)$  is a systematic parameter, which is determined on the development data set; it is used to adjust the tradeoff between local compatibility and global consistence. It is experimentally set to 0.8 in our work.

From Formula 1, we can see that: 1) our method considers the mention-entity similarity, entity-entity similarity and mention-mention similarity. Mention-entity similarity is used to model local compatibility, while entity-entity similarity and mention-mention similarity combined are to model global consistence; and 2) our method prefers configurations where similar mentions have similar entities and with high local compatibility.

$C(\vec{M})$  is worth of more discussion here. It represents the search space, which can be generated using the entity variation list. To achieve this, we first build an inverted index of all entity variation lists, with each unique variation as an entry pointing to a list of entities. Then for any mention  $m$ , we look up the index, and get all possible entities, denoted by  $C(m)$ . In this way, given a mention sequence  $\vec{M} =$

$(m_1, m_2, \dots, m_n)$ , we can enumerate all possible entity sequence  $\vec{E} = (e_1, e_2, \dots, e_n)$ , where  $e_i \in C(m_i)$ . This means  $|C(\vec{M})| = \prod_{m \in M} |C(m)|$ , which is often large. There is one special case: if  $m$  is an OOV mention, i.e.,  $|C(m)| = 0$ , then  $|C(\vec{M})| = 0$ , and we get no solution. To address this problem, we can generate a list of candidates for an OOV mention using its similar mentions. Let  $S(m)$  denote OOV mention  $m$ 's similar mentions, we define  $C(m) = \bigcup_{m' \in S(m)} C(m')$ . If still  $C(m) = 0$ , we remove  $m$  from  $\vec{M}$ , and report we cannot map it to any entity.

Here is an example to illustrate our framework. Suppose we have the following tweets:

- UserA: Yeaahhgg #habemusfut..  
I love monday night futbol =>  
#EnglishPremierLeague ManU vs  
Liverpool<sub>1</sub>
- UserA: Manchester United 3 - Liverpool<sub>2</sub>  
2 #EnglishPremierLeague GLORY, GLORY,  
MAN.UNITED!
- ...

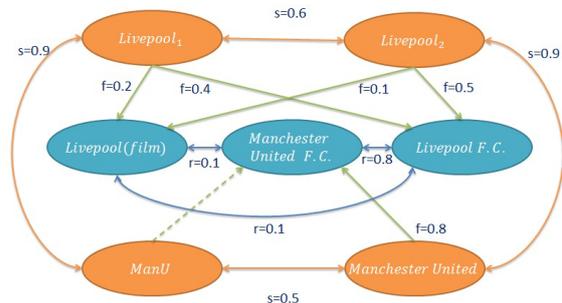


Figure 1: An illustrative example to show our framework. Ovals in orange and in blue represent mentions and entities, respectively. Each mention pair, entity pair, and mention entity pair have a similarity score represented by  $s$ ,  $r$  and  $f$ , respectively.

We need find out the best entity sequence  $\vec{E}^*$  for mentions  $\vec{M} = \{ \text{“Liverpool}_1\text{”}, \text{“Manchester United”}, \text{“ManU”}, \text{“Liverpool}_2\text{”} \}$ , from the entity sequences  $C(\vec{M}) = \{ (\text{Liverpool (film), Manchester United F.C., Manchester United F.C., Liverpool (film)}), \dots, (\text{Liverpool, F.C., Manchester United, F.C., Manchester United F.C., Liverpool (film)}) \}$ . Figure 1 illustrate our solution, where “Liverpool<sub>1</sub>” (on the left) and “Liverpool<sub>2</sub>” (on the right) are linked

to “Liverpool F.C.” (the football club), and “Manchester United” and “ManU” are linked to “Manchester United F.C.”. Notably, “ManU” is an OOV mention, but has a similar mention “Manchester United”, with which “ManU” is successfully mapped.

## 4.2 Features

We group features into three categories: local features related to mention-entity similarity ( $\vec{f}(e, m)$ ), features related to entity-entity similarity ( $r(e_i, e_j)$ ), and features related to mention-mention similarity ( $s(m_i, m_j)$ ).

### 4.2.1 Local Features

- Prior Probability:

$$f_1(m_i, e_i) = \frac{\text{count}(e_i)}{\sum_{\forall e_k \in C(m_i)} \text{count}(e_k)} \quad (2)$$

where  $\text{count}(e)$  denotes the frequency of entity  $e$  in Wikipedia’s anchor texts.

- Context Similarity:

$$f_2(m_i, e_i) = \frac{\text{cocurrence\_number}}{\text{tweet\_length}} \quad (3)$$

where: *cocurrence\_number* is the the number of the words that occur in both the tweet containing  $m_i$  and the Wikipedia page of  $e_i$ ; *tweet\_length* denotes the number of tokens of the tweet containing mention  $m_i$ .

- Edit Distance Similarity:

If  $\text{Length}(m_i) + \text{ED}(m_i, e_i) = \text{Length}(e_i)$ ,  $f_3(m_i, e_i) = 1$ , otherwise 0.  $\text{ED}(\cdot, \cdot)$  computes the character level edit distance. This feature helps to detect whether a mention is an abbreviation of its corresponding entity<sup>7</sup>.

- Mention Contains Title: If the mention contains the entity title, namely the title of the Wikipedia page introducing the entity  $e_i$ ,  $f_4(m_i, e_i) = 1$ , else 0.

- Title Contains Mention: If the entry title contains the mention,  $f_5(m_i, e_i) = 1$ , otherwise 0.

<sup>7</sup>Take “ms” and “Microsoft” for example. The length of “ms” is 2, and the edit distance between them is 7. 2 plus 7 equals to 9, which is the length of “Microsoft”.

### 4.2.2 Features Related to Entity Similarity

There are two representative definitions of entity similarity: in-link based similarity (Milne and Witten, 2008) and category based similarity (Shen et al., 2012). Considering that the Wikipedia categories are often noisy (Milne and Witten, 2008), we adopt in-link based similarity, as defined in Formula 4:

$$r(e_i, e_j) = \frac{\log|g(e_i) \cap g(e_j)| - \log \max(|g(e_i)|, |g(e_j)|)}{\log(\text{Total}) - \log \min(|g(e_i)|, |g(e_j)|)} \quad (4)$$

Where:

- *Total* is the total number of knowledge base entities;
- $g(e)$  is the number of Wikipedia definition pages that have a link to entity  $e$ .

### 4.2.3 Features Related to Mention Similarity

We define 5 features to model the similarity between two mentions  $m_i$  and  $m_j$ , as listed below, where  $t(m)$  denotes the tweet that contains mention  $m$ :

- $s_1(m_i, m_j)$ : The cosine similarity of  $t(m_i)$  and  $t(m_j)$ ; and tweets are represented as TF-IDF vectors;
- $s_2(m_i, m_j)$ : The cosine similarity of  $t(m_i)$  and  $t(m_j)$ ; and tweets are represented as topic distribution vectors;
- $s_3(m_i, m_j)$ : Whether  $t(m_i)$  and  $t(m_j)$  are published by the same account;
- $s_4(m_i, m_j)$ : Whether  $t(m_i)$  and  $t(m_j)$  contain any common hash tag;
- $s_5(m_i, m_j)$ : Edit distance related similarity between  $m_i$  and  $m_j$ , as defined in Formula 5.

$$s_5(m_i, m_j) = 1, \text{ if } \min\{\text{Length}(m_i), \text{Length}(m_j)\} + \text{ED}(m_i, m_j) = \max\{\text{Length}(m_i), \text{Length}(m_j)\},$$

$$\text{else } s_5(m_i, m_j) = 1 - \frac{\text{ED}(m_i, m_j)}{\max\{\text{Length}(m_i), \text{Length}(m_j)\}} \quad (5)$$

Note that: 1) before computing TF-IDF vectors, stop words are removed; 2) we use the Stanford Topic Modeling Toolbox<sup>8</sup> to compute the topic model, and experimentally set the number of topics to 50.

<sup>8</sup><http://nlp.stanford.edu/software/tmt/tmt-0.4/>

Finally, Formula 6 is used to integrate all the features.  $\vec{a} = (a_1, a_2, a_3, a_4, a_5)$  is the feature weight vector for mention similarity, where  $a_k \in (0, 1)$ ,  $k = 1, 2, 3, 4, 5$ , and  $\sum_{k=1}^5 a_k = 1$ .

$$s(m_i, m_j) = \sum_{k=1}^5 a_k s_k(m_i, m_j) \quad (6)$$

### 4.3 Training and Decoding

Given  $n$  mentions  $m_1, m_2, \dots, m_n$  and their corresponding entities  $e_1, e_2, \dots, e_n$ , the goal of training is to determine:  $\vec{w}^*$ , the weights of local features, and  $\vec{a}^*$ , the weights of the features related to mention similarity, according to Formula 7<sup>9</sup>.

$$(\vec{w}^*, \vec{a}^*) = \arg \min_{\vec{w}, \vec{a}} \left\{ \frac{1}{n} \sum_{i=1}^n L_1(e_i, m_i) + \alpha_1 \|\vec{w}\|^2 + \frac{\alpha_2}{2} \sum_{i,j=1}^n s(m_i, m_j) L_2(\vec{a}, e_i, e_j) \right\} \quad (7)$$

Where:

- $L_1$  is the loss function related to local compatibility, which is defined as  $\frac{1}{\vec{w} \cdot \vec{f}(e_i, m_i) + 1}$ ;
- $L_2(\vec{a}, e_i, e_j)$  is the loss function related to global coherence, which is defined as  $\frac{1}{r(e_i, e_j) \sum_{k=1}^5 a_k s_k(m_i, m_j) + 1}$ ;
- $\alpha_1$  is the weight of regularization, which is experimentally set to 1.0;
- $\alpha_2$  is the weight of  $L_2$  loss, which is experimentally set to 0.2.

Since the decoding problem defined by Formula 1 is NP hard (Kulkarni et al., 2009), we develop a greedy hill-climbing approach to tackle this challenge, as demonstrated in Algorithm 1.

In Algorithm 1,  $it$  is the number of iterations;  $Score(\vec{E}, \vec{M}) = \lambda \sum_{i=1}^n \vec{w} \cdot \vec{f}(e_i, m_i) + (1 - \lambda) \sum_{i \neq j} r(e_i, e_j) s(m_i, m_j)$ ;  $\vec{E}_{ij}$  is the vector after replacing  $e_i$  with  $e_j \in C(m_i)$  for current  $\vec{E}$ ;  $sc_{ij}$  is the score of  $\vec{E}_{ij}$ , i.e.,  $Score(\vec{E}_{ij}, \vec{M})$ . In each iteration, this rounding solution iteratively substitute entry  $e_i$  in  $\vec{E}$  to increase the total score  $cur$ . If the score cannot be further improved, it stops and returns current  $\vec{E}$ .

<sup>9</sup>This optimization problem is non-convex. We use coordinate descent to get a local optimal solution.

---

### Algorithm 1 Decoding Algorithm.

---

**Input:** Mention Set  $\vec{M} = (m_1, m_2, \dots, m_n)$

**Output:** Entity Set  $\vec{E} = (e_1, e_2, \dots, e_n)$

```

1: for  $i = 1$  to  $n$  do
2:   Initialize  $e_i^{(0)}$  as the entity with the largest prior
   probability given mention  $m_i$ .
3: end for
4:  $cur = Score(\vec{E}^{(0)}, \vec{M})$ 
5:  $it = 1$ 
6: while true do
7:   for  $i = 1$  to  $n$  do
8:     for  $e_j \in C(m_i)$  do
9:       if  $e_j \neq e_i^{(it-1)}$  then
10:         $\vec{E}_{ij}^{(it)} = \vec{E}^{(it-1)} - \{e_i^{(it-1)}\} + \{e_j\}$ .
11:       end if
12:         $sc_{ij} = Score(\vec{E}_{ij}^{(it)}, \vec{M})$ .
13:       end for
14:     end for
15:      $(l, m) = \operatorname{argmax}_{(i,j)} sc_{ij}$ .
16:      $sc^* = sc_{lm}$ 
17:     if  $sc^* > cur$  then
18:        $cur = sc^*$ .
19:        $\vec{E}^{(it)} = \vec{E}^{(it-1)} - \{e_l^{(it-1)}\} + \{e_m\}$ .
20:        $it = it + 1$ .
21:     else
22:       break
23:     end if
24:   end while
25: return  $\vec{E}^{(it)}$ .

```

---

## 5 Experiments

In this section, we introduce the data set and experimental settings, and present results.

### 5.1 Data Preparation

Following most existing studies, we choose Wikipedia as our knowledge base<sup>10</sup>. We index the Wikipedia definition pages, and prepare all required prior knowledge, such as  $count(e)$ ,  $g(e)$ , and entity variation lists. We also build an inverted index with about 60 million entries for the entity variation lists.

For tweets, we use the data set shared by Meij et al. (2012)<sup>11</sup>. This data set is annotated manually by two volunteers. We get 502 annotated tweets from this data set. We keep 55 of them for

<sup>10</sup>We download the December 2012 version of Wikipedia, which contains about four million articles.

<sup>11</sup><http://ilps.science.uva.nl/resources/wsdm2012-adding-semantics-to-microblog-posts/>.

development, and the remaining for 5 fold cross-validation.

## 5.2 Settings

We consider following settings to evaluate our method.

- Comparing our method with two baselines, i.e., Wikify! (Mihalcea and Csomai, 2007) and the system proposed by Meij et al. (2012)<sup>12</sup>;
- Using only local features;
- Using various mention similarity features;
- Experiments on OOV mentions.

## 5.3 Results

Table 1 reports the comparison results. Our method outperforms both systems in terms of all metrics. Since the main difference between our method and the baselines is that our method considers not only local features, but also global features related to entity similarity and mention similarity, these results indicate the effectiveness of collective inference and global features. For example, we find two baselines incorrectly link “Nickelodeon” in the tweet “BOH will make a special appearance on Nickelodeon’s ‘Yo Gabba Gabba’ tomorrow” to the theater instead of a TV channel. In contrast, our method notices that “Yo Gabba Gabba” in the same tweet can be linked to “Yo Gabba Gabba (TV show)”, and thus it correctly maps “Nickelodeon” to “Nickelodeon (TV channel)”.

System	Pre.	Rec.	F1
Wikify!	0.375	0.421	0.396
Meij’s Method	0.734	0.632	0.679
Our Method	0.752	0.675	0.711

Table 1: Comparison with Baselines.

Table 2 shows the results when local features are incrementally added. It can be seen that: 1) using only Prior Probability feature already yields a reasonable F1; and 2) Context Similarity and Edit Distance Similarity feature have little contribution to the F1, while Mention and Entity Title Similarity feature greatly boosts the F1.

<sup>12</sup>We re-implement Wikify! since we use a new evaluation data set.

Local Feature	Pre.	Rec.	F1
P.P.	0.700	0.599	0.646
+C.S.	0.694	0.597	0.642
+E.D.S.	0.696	0.598	0.643
+M.E.T.S.	0.735	0.632	0.680

Table 2: Local Feature Analysis. P.P., C.S., E.D.S., and M.E.T.S. denote Prior Probability, Context Similarity, Edit Distance Similarity, and Mention and Entity Title Similarity, respectively.

The performance of our method with various mention similarity features is reported in Table 3. First, we can see that with this kind of features, the F1 can be significantly improved from 0.680 to 0.704. Second, we notice that TF-IDF ( $s_1$ ) and Topic Model ( $s_2$ ) features perform equally well, and combining all mention similarity features yields the best performance.

Global Feature	Pre.	Rec.	F1
$s_3+s_4+s_5$	0.744	0.653	0.700
$s_3+s_4+s_5+s_1$	0.759	0.652	0.702
$s_3+s_4+s_5+s_2$	0.760	0.653	0.703
$s_3+s_4+s_5+s_1+s_2$	0.764	0.653	0.704

Table 3: Mention Similarity Feature Analysis.

For any OOV mention, we use the strategy of guessing its possible entity candidates using similar mentions, as discussed in Section 4.1. Table 4 shows the performance of our system for OOV mentions. It can be seen that with our OOV strategy, the recall is improved from 0.653 to 0.675 (with  $p < 0.05$ ) while the Precision is slightly dropped and the overall F1 still gets better. A further study reveals that among all the 125 OOV mentions, there are 48 for which our method cannot find any entity; and nearly half of these 48 OOV mentions do have corresponding entities<sup>13</sup>. This suggests that we may need enlarge the size of variation lists or develop some mention normalization techniques.

OOV Method	Precision	Recall	F1
Ignore OOV Mention	0.764	0.653	0.704
+ OOV Method	0.752	0.675	0.711

Table 4: Performance for OOV Mentions.

<sup>13</sup>“NATO-ukraine cooperations” is such an example. It is mapped to NULL but actually has a corresponding entity “Ukraine-NATO relations”

## 6 Conclusions and Future work

We have presented a collective inference method that jointly links a set of tweet mentions to their corresponding entities. One distinguished characteristic of our method is that it integrates mention-entity similarity, entity-entity similarity, and mention-mention similarity, to address the information lack in a tweet and rich OOV mentions. We evaluate our method on a public data set. Experimental results show our method outperforms two baselines, and suggests the effectiveness of modeling mention-mention similarity, particularly for OOV mention linking.

In the future, we plan to explore two directions. First, we are going to enlarge the size of entity variation lists. Second, we want to integrate the entity mention normalization techniques as introduced by Liu et al. (2012).

## Acknowledgments

We thank the anonymous reviewers for their valuable comments. We also thank all the QuickView team members for the helpful discussions.

## References

- S. Dill, N. Eiron, D. Gibson, D. Gruhl, and R. Guha. 2003. Semtag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th international conference on World Wide Web, WWW '03*, pages 178–186, New York, NY, USA. ACM.
- Maxim Grinev, Maria Grineva, Alexander Boldakov, Leonid Novak, Andrey Syssoev, and Dmitry Lizorkin. 2009. Sifting micro-blogging stream for events of user interest. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '09*, pages 837–837, New York, NY, USA. ACM.
- Xianpei Han and Jun Zhao. 2009. Nlpr-kbp in tac 2009 kbp track: A two-stage method to entity linking. In *Proceedings of Text Analysis Conference*.
- Xianpei Han and Jun Zhao. 2010. Structural semantic relatedness: a knowledge-based method to named entity disambiguation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.
- Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: A graph-based method. In *SIGIR'11*.
- Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–465.
- Xiaohua Liu, Ming Zhou, Xiangyang Zhou, Zhongyang Fu, and Furu Wei. 2012. Joint inference of named entity recognition and normalization for tweets. In *ACL (1)*, pages 526–535.
- Michael Mathioudakis and Nick Koudas. 2010. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, SIGMOD '10*, pages 1155–1158, New York, NY, USA. ACM.
- Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*.
- Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, CIKM '07*, pages 233–242, New York, NY, USA. ACM.
- David Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*.
- Kevin Dela Rosa, Rushin Shah, Bo Lin, Anatole Gershman, and Robert Frederking. 2010. Topical clustering of tweets. In *SWSM'10*.
- Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2012. Liege: Link entities in web lists with knowledge base. In *KDD'12*.
- Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11*, pages 793–803, Stroudsburg, PA, USA. Association for Computational Linguistics.

# Identification of Speakers in Novels

Hua He<sup>†</sup>

Denilson Barbosa<sup>‡</sup>

Grzegorz Kondrak<sup>‡</sup>

<sup>†</sup>Department of Computer Science  
University of Maryland  
huah@cs.umd.edu

<sup>‡</sup>Department of Computing Science  
University of Alberta  
{denilson, gkondrak}@ualberta.ca

## Abstract

Speaker identification is the task of attributing utterances to characters in a literary narrative. It is challenging to automate because the speakers of the majority of utterances are not explicitly identified in novels. In this paper, we present a supervised machine learning approach for the task that incorporates several novel features. The experimental results show that our method is more accurate and general than previous approaches to the problem.

## 1 Introduction

Novels are important as social communication documents, in which novelists develop the plot by means of discourse between various characters. In spite of a frequently expressed opinion that all novels are simply variations of a certain number of basic plots (Tobias, 2012), every novel has a unique plot (or several plots) and a different set of characters. The interactions among characters, especially in the form of conversations, help the readers construct a mental model of the plot and the changing relationships between characters. Many of the complexities of interpersonal relationships, such as romantic interests, family ties, and rivalries, are conveyed by utterances.

A precondition for understanding the relationship between characters and plot development in a novel is the identification of speakers behind all utterances. However, the majority of utterances are not explicitly tagged with speaker names, as is the case in stage plays and film scripts. In most cases, authors rely instead on the readers' comprehension of the story and of the differences between characters.

Since manual annotation of novels is costly, a system for automatically determining speakers of utterances would facilitate other tasks related to

the processing of literary texts. Speaker identification could also be applied on its own, for instance in generating high quality audio books without human lecturers, where each character would be identifiable by a distinct way of speaking. In addition, research on spoken language processing for broadcast and multi-party meetings (Salamin et al., 2010; Favre et al., 2009) has demonstrated that the analysis of dialogues is useful for the study of social interactions.

In this paper, we investigate the task of speaker identification in novels. Departing from previous approaches, we develop a general system that can be trained on relatively small annotated data sets, and subsequently applied to other novels for which no annotation is available. Since every novel has its own set of characters, speaker identification cannot be formulated as a straightforward tagging problem with a universal set of fixed tags. Instead, we adopt a ranking approach, which enables our model to be applied to literary texts that are different from the ones it has been trained on.

Our approach is grounded in a variety of features that are easily generalizable across different novels. Rather than attempt to construct complete semantic models of the interactions, we exploit lexical and syntactic clues in the text itself. We propose several novel features, including the speaker alternation pattern, the presence of vocatives in utterances, and unsupervised actor-topic features that associate speakers with utterances on the basis of their content. Experimental evaluation shows that our approach not only outperforms the baseline, but also compares favorably to previous approaches in terms of accuracy and generality, even when tested on novels and authors that are different from those used for training.

The paper is organized as follows. After discussing previous work, and defining the terminology, we present our approach and the features that it is based on. Next, we describe the data, the an-

notation details, and the results of our experimental evaluation. At the end, we discuss an application to extracting a set of family relationships from a novel.

## 2 Related Work

Previous work on speaker identification includes both rule-based and machine-learning approaches. Glass and Bangay (2007) propose a rule generalization method with a scoring scheme that focuses on the speech verbs. The verbs, such as *said* and *cried*, are extracted from the communication category of WordNet (Miller, 1995). The speech-verb-actor pattern is applied to the utterance, and the speaker is chosen from the available candidates on the basis of a scoring scheme. Sarmiento and Nunes (2009) present a similar approach for extracting speech quotes from online news texts. They manually define 19 variations of frequent speaker patterns, and identify a total of 35 candidate speech verbs. The rule-based methods are typically characterized by low coverage, and are too brittle to be reliably applied to different domains and changing styles.

Elson and McKeown (2010) (henceforth referred to as EM2010) apply the supervised machine learning paradigm to a corpus of utterances extracted from novels. They construct a single feature vector for each pair of an utterance and a speaker candidate, and experiment with various WEKA classifiers and score-combination methods. To identify the speaker of a given utterance, they assume that all previous utterances are already correctly assigned to their speakers. Our approach differs in considering the utterances in a sequence, rather than independently from each other, and in removing the unrealistic assumption that the previous utterances are correctly identified.

The speaker identification task has also been investigated in other domains. Bethard et al. (2004) identify opinion holders by using semantic parsing techniques with additional linguistic features. Pouliquen et al. (2007) aim at detecting direct speech quotations in multilingual news. Krestel et al. (2008) automatically tag speech sentences in newspaper articles. Finally, Ruppenhofer et al. (2010) implement a rule-based system to enrich German cabinet protocols with automatic speaker attribution.

## 3 Definitions and Conventions

In this section, we introduce the terminology used in the remainder of the paper. Our definitions are different from those of EM2010 partly because we developed our method independently, and partly because we disagree with some of their choices. The examples are from Jane Austen’s *Pride and Prejudice*, which was the source of our development set.

An *utterance* is a connected text that can be attributed to a single speaker. Our task is to associate each utterance with a single speaker. Utterances that are attributable to more than one speaker are rare; in such cases, we accept correctly identifying one of the speakers as sufficient. In some cases, an utterance may include more than one quotation-delimited sequence of words, as in the following example.

*“Miss Bingley told me,” said Jane, “that he never speaks much.”*

In this case, the words *said Jane* are simply a speaker tag inserted into the middle of the quoted sentence. Unlike EM2010, we consider this a single utterance, rather than two separate ones.

We assume that all utterances within a paragraph can be attributed to a single speaker. This “one speaker per paragraph” property is rarely violated in novels — we identified only five such cases in *Pride & Prejudice*, usually involving one character citing another, or characters reading letters containing quotations. We consider this an acceptable simplification, much like assigning a single part of speech to each word in a corpus. We further assume that each utterance is contained within a single paragraph. Exceptions to this rule can be easily identified and resolved by detecting quotation marks and other typographical conventions.

The paragraphs without any quotations are referred to as *narratives*. The term *dialogue* denotes a series of utterances together with related narratives, which provide the context of conversations. We define a dialogue as a series of utterances and intervening narratives, with no more than three continuous narratives. The rationale here is that more than three narratives without any utterances are likely to signal the end of a particular dialogue.

We distinguish three types of utterances, which are listed with examples in Table 1: *explicit speaker* (identified by name within the paragraph),

Category	Example
Implicit speaker	“ <i>Don’t keep coughing so, Kitty, for heaven’s sake!</i> ”
Explicit speaker	“ <i>I do not cough for my own amusement,</i> ” replied <b>Kitty</b> .
Anaphoric speaker	“ <i>Kitty has no discretion in her coughs,</i> ” said <b>her father</b> .

Table 1: Three types of utterances.

*anaphoric speaker* (identified by an anaphoric expression), and *implicit speaker* (no speaker information within the paragraph). Typically, the majority of utterances belong to the implicit-speaker category. In *Pride & Prejudice* only roughly 25% of the utterances have explicit speakers, and an even smaller 15% belong to the anaphoric-speaker category. In modern fiction, the percentage of explicit attributions is even lower.

## 4 Speaker Identification

In this section, we describe our method of extracting explicit speakers, and our ranking approach, which is designed to capture the speaker alternation pattern.

### 4.1 Extracting Speakers

We extract explicit speakers by focusing on the speech verbs that appear before, after, or between quotations. The following verbs cover most cases in our development data: *say, speak, talk, ask, reply, answer, add, continue, go on, cry, sigh, and think*. If a verb from the above short list cannot be found, any verb that is preceded by a name or a personal pronoun in the vicinity of the utterance is selected as the speech verb.

In order to locate the speaker’s name or anaphoric expression, we apply a deterministic method based on syntactic rules. First, all paragraphs that include narrations are parsed with a dependency parser. For example, consider the following paragraph:

*As they went downstairs together, Charlotte said, “I shall depend on hearing from you very often, Eliza.”*

The parser identifies a number of dependency relations in the text, such as *doobj(went-3, downstairs-4)* and *advmod(went-3, together-5)*. Our method extracts the speaker’s name from the dependency relation *nsbj(said-8, Charlotte-7)*, which links a

speech verb with a noun phrase that is the syntactic subject of a clause.

Once an explicit speaker’s name or an anaphoric expression is located, we determine the corresponding gender information by referring to the character list or by following straightforward rules to handle the anaphora. For example, if the utterance is followed by the phrase *she said*, we infer that the gender of the speaker is female.

### 4.2 Ranking Model

In spite of the highly sequential nature of the chains of utterances, the speaker identification task is difficult to model as sequential prediction. The principal problem is that, unlike in many NLP problems, a general fixed tag set cannot be defined beyond the level of an individual novel. Since we aim at a system that could be applied to any novel with minimal pre-processing, sequential prediction algorithms such as Conditional Random Fields are not directly applicable.

We propose a more flexible approach that assigns scores to candidate speakers for each utterance. Although the sequential information is not directly modeled with tags, our system is able to indirectly utilize the speaker alternation pattern using the method described in the following section. We implement our approach with *SVM-rank* (Joachims, 2006).

### 4.3 Speaker Alternation Pattern

The speaker alternation pattern is often employed by authors in dialogues between two characters. After the speakers are identified explicitly at the beginning of a dialogue, the remaining odd-numbered and even-numbered utterances are attributable to the first and second speaker, respectively. If one of the speakers “misses their turn”, a clue is provided in the text to reset the pattern.

Based on the speaker alternation pattern, we make the following two observations:

1. The speakers of consecutive utterances are usually different.
2. The speaker of the  $n$ -th utterance in a dialogue is likely to be the same as the speaker of the  $(n - 2)$ -th utterance.

Our ranking model incorporates the speaker alternation pattern by utilizing a feature expansion scheme. For each utterance  $n$ , we first generate its own features (described in Section 5), and

Features	Novelty
Distance to Utterance	No
Speaker Appearance Count	No
Speaker Name in Utterance	No
Unsupervised Actor-Topic Model	Yes
Vocative Speaker Name	Yes
Neighboring Utterances	Yes
Gender Matching	Yes
Presence Matching	Yes

Table 2: Principal feature sets.

subsequently we add three more feature sets that represent the following neighboring utterances:  $n - 2$ ,  $n - 1$  and  $n + 1$ . Informally, the features of the utterances  $n - 1$  and  $n + 1$  encode the first observation, while the features representing the utterance  $n - 2$  encode the second observation. In addition, we include a set of four binary features that are set for the utterances in the range  $[n - 2, n + 1]$  if the corresponding explicit speaker matches the candidate speaker of the current utterance.

## 5 Features

In this section, we describe the set of features used in our ranking approach. The principal feature sets are listed in Table 2, together with an indication whether they are novel or have been used in previous work.

### 5.1 Basic Features

A subset of our features correspond to the features that were proposed by EM2010. These are mostly features related to speaker names. For example, since names of speakers are often mentioned in the vicinity of their utterances, we count the number of words separating the utterance and a name mention. However, unlike EM2010, we consider only the two nearest characters in each direction, to reflect the observation that speakers tend to be mentioned by name immediately before or after their corresponding utterances. Another feature is used to represent the number of appearances for speaker candidates. This feature reflects the relative importance of a given character in the novel. Finally, we use a feature to indicate the presence or absence of a candidate speaker’s name within the utterance. The intuition is that speakers are unlikely to mention their own name.

Feature	Example
start of utterance	“ <i>Kitty</i> ...
before period	... <i>Jane</i> .
between commas	..., <i>Elizabeth</i> , ...
between comma & period	..., <i>Mrs. Hurst</i> .
before exclamation mark	... <i>Mrs. Bennet!</i>
before question mark	... <i>Lizzy?</i> ...
vocative phrase	<i>Dear</i> ...
after vocative phrase	<i>Oh! Lydia</i> ...
2nd person pronoun	... <i>you</i> ...

Table 3: Features for the vocative identification.

### 5.2 Vocatives

We propose a novel vocative feature, which encodes the character that is explicitly addressed in an utterance. For example, consider the following utterance:

“*I hope Mr. Bingley will like it, Lizzy.*”

Intuitively, the speaker of the utterance is neither *Mr. Bingley* nor *Lizzy*; however, the speaker of the next utterance is likely to be *Lizzy*. We aim at capturing this intuition by identifying the addressee of the utterance.

We manually annotated vocatives in about 900 utterances from the training set. About 25% of the names within utterance were tagged as vocatives. A Logistic Regression classifier (Agresti, 2006) was trained to identify the vocatives. The classifier features are shown in Table 3. The features are designed to capture punctuation context, as well as the presence of typical phrases that accompany vocatives. We also incorporate interjections like “oh!” and fixed phrases like “my dear”, which are strong indicators of vocatives. Under 10-fold cross validation, the model achieved an F-measure of 93.5% on the training set.

We incorporate vocatives in our speaker identification system by means of three binary features that correspond to the utterances  $n - 1$ ,  $n - 2$ , and  $n - 3$ . The features are set if the detected vocative matches the candidate speaker of the current utterance  $n$ .

### 5.3 Matching Features

We incorporate two binary features for indicating the gender and the presence of a candidate speaker. The gender matching feature encodes the gender agreement between a speaker candidate and the speaker of the current utterance. The gender information extraction is applied to two utterance

groups: the anaphoric-speaker utterances, and the explicit-speaker utterances. We use the technique described in Section 4.1 to determine the gender of a speaker of the current utterance. In contrast with EM2010, this is not a hard constraint.

The presence matching feature indicates whether a speaker candidate is a likely participant in a dialogue. Each dialogue consists of continuous utterance paragraphs together with neighboring narration paragraphs as defined in Section 3. The feature is set for a given character if its name or alias appears within the dialogue.

#### 5.4 Unsupervised Actor-Topic Features

The final set of features is generated by the unsupervised actor-topic model (ACTM) (Celikyilmaz et al., 2010), which requires no annotated training data. The ACTM, as shown in Figure 1, extends the work of author-topic model in (Rosen-Zvi et al., 2010). It can model dialogues in a literary text, which take place between two or more speakers conversing on different topics, as distributions over topics, which are also mixtures of the term distributions associated with multiple speakers. This follows the linguistic intuition that rich contextual information can be useful in understanding dialogues.

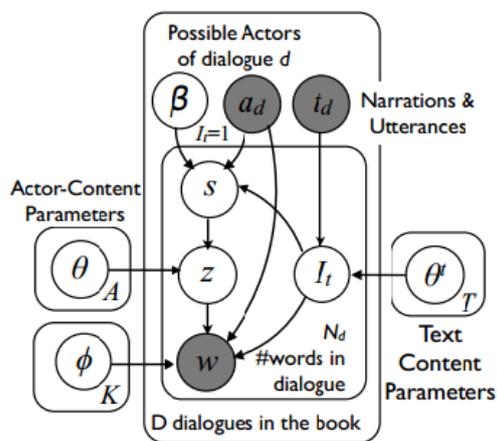


Figure 1: Graphical Representation of ACTM.

The ACTM predicts the most likely speakers of a given utterance by considering the content of an utterance and its surrounding contexts. The Actor-Topic-Term probabilities are calculated by using both the relationship of utterances and the surrounding textual clues. In our system, we utilize four binary features that correspond to the four top ranking positions from the ACTM model.

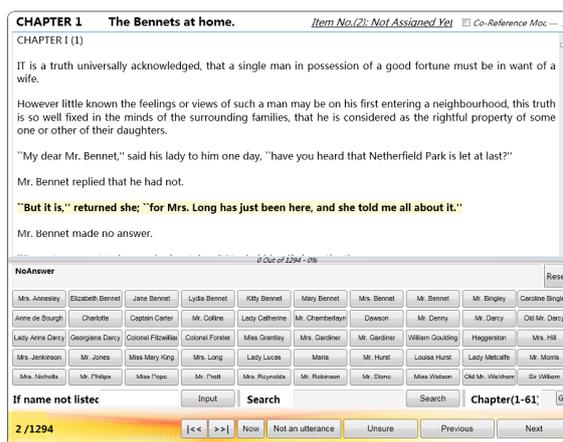


Figure 2: Annotation Tool GUI.

## 6 Data

Our principal data set is derived from the text of *Pride and Prejudice*, with chapters 19–26 as the test set, chapters 27–33 as the development set, and the remaining 46 chapters as the training set. In order to ensure high-quality speaker annotations, we developed a graphical interface (Figure 2), which displays the current utterance in context, and a list of characters in the novel. After the speaker is selected by clicking a button, the text is scrolled automatically, with the next utterance highlighted in yellow. The complete novel was annotated by a student of English literature. The annotations are publicly available<sup>1</sup>.

For the purpose of a generalization experiment, we also utilize a corpus of utterances from the 19th and 20th century English novels compiled by EM2010. The corpus differs from our data set in three aspects. First, as discussed in Section 3, we treat all quoted text within a single paragraph as a single utterance, which reduces the total number of utterances, and results in a more realistic reporting of accuracy. Second, our data set includes annotations for all utterances in the novel, as opposed to only a subset of utterances from several novels, which are not necessarily contiguous. Lastly, our annotations come from a single expert, while the annotations in the EM2010 corpus were collected through Amazon’s Mechanical Turk, and filtered by voting. For example, out of 308 utterances from *The Steppe*, 244 are in fact annotated, which raises the question whether the discarded utterances tend to be more difficult to annotate.

Table 4 shows the number of utterances in all

<sup>1</sup>[www.cs.ualberta.ca/~kondrak/austen](http://www.cs.ualberta.ca/~kondrak/austen)

	IS	AS	ES	Total
<i>Pride &amp; P.</i> (all)	663	292	305	1260
<i>Pride &amp; P.</i> (test)	65	29	32	126
<i>Emma</i>	236	55	106	397
<i>The Steppe</i>	93	39	112	244

Table 4: The number of utterances in various data sets by the type (IS - Implicit Speaker; AS - Anaphoric Speaker; ES - Explicit Speaker).

data sets. We selected Jane Austen’s *Emma* as a different novel by the same author, and Anton Chekhov’s *The Steppe* as a novel by a different author for our generalization experiments.

Since our goal is to match utterances to characters rather than to name mentions, a preprocessing step is performed to produce a list of characters in the novel and their aliases. For example, *Elizabeth Bennet* may be referred to as *Liz*, *Lizzy*, *Miss Lizzy*, *Miss Bennet*, *Miss Eliza*, and *Miss Elizabeth Bennet*. We apply a name entity tagger, and then group the names into sets of character aliases, together with their gender information. The sets of aliases are typically small, except for major characters, and can be compiled with the help of web resources, such as Wikipedia, or study guides, such as CliffsNotes<sup>TM</sup>. This preprocessing step could also be performed automatically using a canonicalization method (Andrews et al., 2012); however, since our focus is on speaker identification, we decided to avoid introducing annotation errors at this stage.

Other preprocessing steps that are required for processing a new novel include standardizing the typographical conventions, and performing POS tagging, NER tagging, and dependency parsing. We utilize the Stanford tools (Toutanova et al., 2003; Finkel et al., 2005; Marneffe et al., 2006).

## 7 Evaluation

In this section, we describe experiments conducted to evaluate our speaker identification approach. We refer to our main model as NEIGHBORS, because it incorporates features from the neighboring utterances, as described in Section 4.3. In contrast, the INDIVIDUAL model relies only on features from the current utterance. In an attempt to reproduce the evaluation methodology of EM2010, we also test the ORACLE model, which has access to the gold-standard information about the speakers of eight neighboring utterances in the

	<i>Pride &amp; P.</i>	<i>Emma</i>	<i>Steppe</i>
BASELINE	42.0	44.1	66.8
INDIVIDUAL	77.8	67.3	74.2
NEIGHBORS	82.5	74.8	80.3
ORACLE	86.5	80.1	83.6

Table 5: Speaker identification accuracy (in %) on *Pride & Prejudice*, *Emma*, and *The Steppe*.

range  $[n - 4, n + 4]$ . Lastly, the BASELINE approach selects the name that is the closest in the narration, which is more accurate than the “most recent name” baseline.

### 7.1 Results

Table 5 shows the results of the models trained on annotated utterances from *Pride & Prejudice* on three test sets. As expected, the accuracy of all learning models on the test set that comes from the same novel is higher than on unseen novels. However, in both cases, the drop in accuracy for the NEIGHBORS model is less than 10%.

Surprisingly, the accuracy is higher on *The Steppe* than on *Emma*, even though the different writing style of Chekhov should make the task more difficult for models trained on Austen’s prose. The protagonists of *The Steppe* are mostly male, and the few female characters rarely speak in the novel. This renders our gender feature virtually useless, and results in lower accuracy on anaphoric speakers than on explicit speakers. On the other hand, Chekhov prefers to mention speaker names in the dialogues (46% of utterances are in the explicit-speaker category), which makes his prose slightly easier in terms of speaker identification.

The relative order of the models is the same on all three test sets, with the NEIGHBORS model consistently outperforming the INDIVIDUAL model, which indicates the importance of capturing the speaker alternation pattern. The performance of the NEIGHBORS model is actually closer to the ORACLE model than to the INDIVIDUAL model.

Table 6 shows the results on *Emma* broken down according to the type of the utterance. Unsurprisingly, the explicit speaker is the easiest category, with nearly perfect accuracy. Both the INDIVIDUAL and the NEIGHBORS models do better on anaphoric speakers than on implicit speakers, which is also expected. However, it is not the

	IS	AS	ES	Total
INDIVIDUAL	52.5	67.3	100.0	67.3
NEIGHBORS	63.1	76.4	100.0	74.8
ORACLE	74.2	69.1	99.1	80.1

Table 6: Speaker identification accuracy (in %) on Austen’s *Emma* by the type of utterance.

case for the ORACLE model. We conjecture that the ORACLE model relies heavily on the neighborhood features (which are rarely wrong), and consequently tends to downplay the gender information, which is the only information extracted from the anaphora. In addition, anaphoric speaker is the least frequent of the three categories.

Table 7 shows the results of an ablation study performed to investigate the relative importance of features. The INDIVIDUAL model serves as the base model from which we remove specific features. All tested features appear to contribute to the overall performance, with the distance features and the unsupervised actor-topic features having the most pronounced impact. We conclude that the incorporation of the neighboring features, which is responsible for the difference between the INDIVIDUAL and NEIGHBORS models, is similar in terms of importance to our strongest textual features.

Feature	Impact
Closest Mention	-6.3
Unsupervised ACTM	-5.6
Name within Utterance	-4.8
Vocative	-2.4

Table 7: Results of feature ablation (in % accuracy) on *Pride & Prejudice*.

## 7.2 Comparison to EM2010

In this section we analyze in more detail our results on *Emma* and *The Steppe* against the published results of the state-of-the-art EM2010 system. Recall that both novels form a part of the corpus that was created by EM2010 for the development of their system.

Direct comparison to EM2010 is difficult because they compute the accuracy separately for seven different categories of utterances. For each category, they experiment with all combinations of three different classifiers and four score combination methods, and report only the accuracy

Character		
<i>id</i>	<i>name</i>	<i>gender</i>
...		
9	Mr. Collins	m
10	Charlotte	f
11	Jane Bennet	f
12	Elizabeth Bennet	f
...		

Relation			
<i>from</i>	<i>to</i>	<i>type</i>	<i>mode</i>
...			
10	9	husband	explicit
9	10	wife	derived
10	12	friend	explicit
12	10	friend	derived
11	12	sister	explicit
...			

Figure 3: Relational database with extracted social network.

achieved by the best performing combination on that category. In addition, they utilize the ground truth speaker information of the preceding utterances. Therefore, their results are best compared against our ORACLE approach.

Unfortunately, EM2010 do not break down their results by novel. They report the overall accuracy of 63% on both “anaphora trigram” (our *anaphoric speaker*), and “quote alone” (similar to our *implicit speaker*). If we combine the two categories, the numbers corresponding to our NEIGHBORS model are 65.6% on *Emma* and 64.4% on *The Steppe*, while ORACLE achieves 73.2% and 70.5%, respectively. Even though a direct comparison is not feasible, the numbers are remarkable considering the context of the experiment, which strongly favors the EM2010 system.

## 8 Extracting Family Relationships

In this section, we describe an application of the speaker identification system to the extraction of family relationships. Elson et al. (2010) extract unlabeled networks where the nodes represent characters and edges indicate their *proximity*, as indicated by their interactions. Our goal is to construct networks in which edges are labeled by the mutual relationships between characters in a novel. We focus on family relationships, but also include social relationships, such as *friend*

```

INSERT INTO Relation (id1, id2, t, m)
  SELECT r.to AS id1, r.from AS id2 , 'wife' AS t, 'derived' AS m
  FROM Relation r
  WHERE r.type='husband' AND r.mode='explicit' AND
        NOT EXISTS(SELECT * FROM Relation r2
                   WHERE r2.from=r.to AND r2.to=r.from AND r2.type=t)

```

Figure 4: An example inference rule.

and *attracted-to*.

Our approach to building a social network from the novel is to build an active database of relationships explicitly mentioned in the text, which is expanded by triggering the execution of queries that deduce implicit relations. This inference process is repeated for every discovered relationship until no new knowledge can be inferred.

The following example illustrates how speaker identification helps in the extraction of social relations among characters. Consider, the following conversation:

“How so? how can it affect them?”  
 “My dear Mr. Bennet,” replied *his wife*,  
 “how can you be so tiresome!”

If the speakers are correctly identified, the utterances are attributed to *Mr. Bennet* and *Mrs. Bennet*, respectively. Furthermore, the second utterance implies that its speaker is the wife of the preceding speaker. This is an example of an explicit relationship which is included in our database. Several similar extraction rules are used to extract explicit mentions indicating family and affective relations, including *mother*, *nephew*, and *fiancee*. We can also derive relationships that are not explicitly mentioned in the text; for example, that *Mr. Bennet* is the husband of *Mrs. Bennet*.

Figure 3 shows a snippet of the relational database of the network extracted from *Pride & Prejudice*. Table *Character* contains all characters in the book, each with a unique identifier and gender information, while Table *Relation* contains all relationships that are explicitly mentioned in the text or derived through reasoning.

Figure 4 shows an example of an inference rule used in our system. The rule derives a new relationship indicating that character  $c_1$  is the *wife* of character  $c_2$  if it is known (through an explicit mention in the text) that  $c_2$  is the *husband* of  $c_1$ . One condition for the rule to be applied is that the database must not already contain a record indicating the wife relationship. This inference rule

would derive the tuple in Figure 3 indicating that the wife of Mr. Collins is Charlotte.

In our experiment with *Pride & Prejudice*, a total of 55 explicitly indicated relationships were automatically identified once the utterances were attributed to the characters. From those, another 57 implicit relationships were derived through inference. A preliminary manual inspection of the set of relations extracted by this method (Makazhanov et al., 2012) indicates that all of them are correct, and include about 40% all personal relations that can be inferred by a human reader from the text of the novel.

## 9 Conclusion and Future Work

We have presented a novel approach to identifying speakers of utterances in novels. Our system incorporates a variety of novel features which utilize vocatives, unsupervised actor-topic models, and the speaker alternation pattern. The results of our evaluation experiments indicate a substantial improvement over the current state of the art.

There are several interesting directions for the future work. Although the approach introduced in this paper appears to be sufficiently general to handle novels written in a different style and period, more sophisticated statistical graphical models may achieve higher accuracy on this task. A reliable automatic generation of characters and their aliases would remove the need for the preprocessing step outlined in Section 6. The extraction of social networks in novels that we discussed in Section 8 would benefit from the introduction of additional inference rules, and could be extended to capture more subtle notions of sentiment or relationship among characters, as well as their development over time.

We have demonstrated that speaker identification can help extract family relationships, but the converse is also true. Consider the following utterance:

“Lizzy,” said her father, “I have given him my consent.”

In order to deduce the speaker of the utterance, we need to combine the three pieces of information: (a) the utterance is addressed to Lizzy (vocative prediction), (b) the utterance is produced by Lizzy’s father (pronoun resolution), and (c) Mr. Bennet is the father of Lizzy (relationship extraction). Similarly, in the task of compiling a list of characters, which involves resolving aliases such as *Caroline*, *Caroline Bingley*, and *Miss Bingley*, simultaneous extraction of family relationships would help detect the ambiguity of *Miss Bennet*, which can refer to any of several sisters. A joint approach to resolving speaker attribution, relationship extraction, co-reference resolution, and alias-to-character mapping would not only improve the accuracy on all these tasks, but also represent a step towards deeper understanding of complex plots and stories.

## Acknowledgments

We would like to thank Asli Celikyilmaz for collaboration in the early stages of this project, Susan Brown and Michelle Di Cintio for help with data annotation, and David Elson for the attempt to compute the accuracy of the EM2010 system on *Pride & Prejudice*. This research was partially supported by the Natural Sciences and Engineering Research Council of Canada.

## References

- Alan Agresti. 2006. Building and applying logistic regression models. In *An Introduction to Categorical Data Analysis*. John Wiley & Sons, Inc.
- Nicholas Andrews, Jason Eisner, and Mark Dredze. 2012. Name phylogeny: A generative model of string variation. In *EMNLP-CoNLL*.
- Steven Bethard, Hong Yu, Ashley Thornton, Vasileios Hatzivassiloglou, and Dan Jurafsky. 2004. Automatic extraction of opinion propositions and their holders. In *AAAI Spring Symposium on Exploring Attitude and Affect in Text*.
- Asli Celikyilmaz, Dilek Hakkani-Tur, Hua He, Grzegorz Kondrak, and Denilson Barbosa. 2010. The actor-topic model for extracting social networks in literary narrative. In *Proceedings of the NIPS 2010 Workshop - Machine Learning for Social Computing*.
- David K. Elson and Kathleen McKeown. 2010. Automatic attribution of quoted speech in literary narrative. In *AAAI*.
- David K. Elson, Nicholas Dames, and Kathleen McKeown. 2010. Extracting social networks from literary fiction. In *ACL*.
- Sarah Favre, Alfred Dielmann, and Alessandro Vinciarelli. 2009. Automatic role recognition in multi-party recordings using social networks and probabilistic sequential models. In *ACM Multimedia*.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*.
- Kevin Glass and Shaun Bangay. 2007. A naive salience-based method for speaker identification in fiction books. In *Proceedings of the 18th Annual Symposium of the Pattern Recognition*.
- Thorsten Joachims. 2006. Training linear SVMs in linear time. In *KDD*.
- Ralf Krestel, Sabine Bergler, and René Witte. 2008. Minding the source: Automatic tagging of reported speech in newspaper articles. In *LREC*.
- Aibek Makazhanov, Denilson Barbosa, and Grzegorz Kondrak. 2012. Extracting family relations from literary fiction. Unpublished manuscript.
- Marie Catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*.
- George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41.
- Bruno Pouliquen, Ralf Steinberger, and Clive Best. 2007. Automatic detection of quotations in multilingual news. In *RANLP*.
- Michal Rosen-Zvi, Chaitanya Chemudugunta, Thomas L. Griffiths, Padhraic Smyth, and Mark Steyvers. 2010. Learning author-topic models from text corpora. *ACM Trans. Inf. Syst.*, 28(1).
- Josef Ruppenhofer, Caroline Sporleder, and Fabian Shirokov. 2010. Speaker attribution in cabinet protocols. In *LREC*.
- Hugues Salamin, Alessandro Vinciarelli, Khiet Truong, and Gelareh Mohammadi. 2010. Automatic role recognition based on conversational and prosodic behaviour. In *ACM Multimedia*.
- Luis Sarmiento and Sergio Nunes. 2009. Automatic extraction of quotes and topics from news feeds. In *4th Doctoral Symposium on Informatics Engineering*.
- Ronald B. Tobias. 2012. *20 Master Plots: And How to Build Them*. Writer’s Digest Books, 3rd edition.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL-HLT*.

# Language Acquisition and Probabilistic Models: keeping it simple

Aline Villavicencio<sup>\*</sup>, Marco Idiart<sup>∇</sup> Robert Berwick<sup>◇</sup>, Igor Malioutov<sup>♣</sup>

<sup>\*</sup>Institute of Informatics, Federal University of Rio Grande do Sul (Brazil)

<sup>∇</sup>Institute of Physics, Federal University of Rio Grande do Sul (Brazil)

<sup>◇</sup>LIDS, Dept. of EECS, Massachusetts Institute of Technology (USA)

<sup>♣</sup>CSAIL, Dept. of EECS, Massachusetts Institute of Technology (USA)

avillavicencio@inf.ufrgs.br, marco.idiart@if.ufrgs.br

berwick@csail.mit.edu, igorm@mit.edu

## Abstract

Hierarchical Bayesian Models (HBMs) have been used with some success to capture empirically observed patterns of under- and overgeneralization in child language acquisition. However, as is well known, HBMs are “ideal” learning systems, assuming access to unlimited computational resources that may not be available to child language learners. Consequently, it remains crucial to carefully assess the use of HBMs along with alternative, possibly simpler, candidate models. This paper presents such an evaluation for a language acquisition domain where explicit HBMs have been proposed: the acquisition of English dative constructions. In particular, we present a detailed, empirically-grounded model-selection comparison of HBMs vs. a simpler alternative based on clustering along with maximum likelihood estimation that we call linear competition learning (LCL). Our results demonstrate that LCL can match HBM model performance without incurring on the high computational costs associated with HBMs.

## 1 Introduction

In recent years, with advances in probability and estimation theory, there has been much interest in Bayesian models (BMs) (Chater, Tenenbaum, and Yuille, 2006; Jones and Love, 2011) and their application to child language acquisition with its challenging com-

ination of structured information and incomplete knowledge, (Perfors, Tenenbaum, and Wonnacott, 2010; Hsu and Chater, 2010; Parisien, Fazly, and Stevenson, 2008; Parisien and Stevenson, 2010) as they offer several advantages in this domain. They can readily handle the evident noise and ambiguity of acquisition input, while at the same time providing efficiency via priors that mirror known pre-existing language biases. Further, hierarchical Bayesian Models (HBMs) can combine distinct abstraction levels of linguistic knowledge, from variation at the level of individual lexical items, to cross-item variation, using hyper-parameters to capture observed patterns of both under- and over-generalization as in the acquisition of e.g. dative alternations in English (Hsu and Chater, 2010; Perfors, Tenenbaum, and Wonnacott, 2010), and verb frames in a controlled artificial language (Wonnacott, Newport, and Tanenhaus, 2008).

HBMs can thus be viewed as providing a “rational” upper bound on language learnability, yielding optimal models that account for observed data while minimizing any required prior information. In addition, the clustering implicit in HBM modeling introduces additional parameters that can be tuned to specific data patterns. However, this comes at a well-known price: HBMs generally are also ideal learning systems, known to be computationally infeasible (Kwisthout, Wareham, and van Rooij, 2011). Approximations proposed to ensure computational tractability, like reducing the number of classes that need to be learned may also be linguistically and cognitively implausible. For instance, in terms of verb learning, this could

take the form of reducing the number of sub-categorization frames to the relevant subset, as in (Perfors, Tenenbaum, and Wonnacott, 2010), where only 2 frames are considered for ‘take’, when in fact it is listed in 6 frames by Levin (1993). Finally, comparison of various Bayesian models of the same task is rare (Jones and Love, 2011) and Bayesian inference generally can be demonstrated as simply one class of regularization or smoothing techniques among many others; given the problem at hand, there may well be other, equally compelling regularization methods for dealing with the bias-variance dilemma (e.g., SVMs (Shalizi, 2009)). Consequently, the relevance of HBMs for cognitively accurate accounts of human learning remains uncertain and needs to be carefully assessed.

Here we argue that the strengths of HBMs for a given task must be evaluated in light of their computational and cognitive costs, and compared to other viable alternatives. The focus should be on finding the simplest statistical models consistent with a given behavior, particularly one that aligns with known cognitive limitations. In the case of many language acquisition tasks this behavior often takes the form of overgeneralization, but with eventual convergence to some target language given exposure to more data.

In particular, in this paper we consider how children acquire English dative verb constructions, comparing HBMs to a simpler alternative, a linear competition learning (LCL) algorithm that models the behavior of a given verb as the linear competition between the evidence for that verb, and the average behavior of verbs belonging to its same class. The results show that combining simple clustering methods along with ordinary maximum likelihood estimation yields a result comparable to HBM performance, providing an alternative account of the same facts, without the computational costs incurred by HBM models that must rely, for example, on Markov Chain Monte Carlo (MCMC) methods for numerically integrating complex likelihood integrals, or on Chinese Restaurant Process (CRP) for producing partitions.

In terms of Marr’s hierarchy (Marr, 1982) learning verb alternations is an abstract com-

putational problem (Marr’s type I), solvable by many type II methods combining representations (models, viz. HBMs or LCLs) with particular algorithms. The HBM convention of adopting ideal learning amounts to invoking unbounded algorithmic resources, solvability in principle, even though in practice such methods, even approximate ones, are provably NP-hard (cf. (Kwisthout, Wareham, and van Rooij, 2011)). Assuming cognitive plausibility as a desideratum, we therefore examine whether HBMs can also be approximated by another type II method (LCLs) that does not demand such intensive computation. Any algorithm that approximates an HBM can be viewed as implementing a somewhat different underlying model; if it replicates HBM prediction performance but is simpler and less computationally complex then we assume it is preferable.

This paper is organized as follows: we start with a discussion of formalizations of language acquisition tasks, §2. We present our experimental framework for the dative acquisition task, formalizing a range of learning models from simple MLE methods to HBM techniques, §3, and a computational evaluation of each model, §4. We finish with conclusions and possibilities for future work, §5.

## 2 Evidence in Language Acquisition

A familiar problem for language acquisition is how children learn which verbs participate in so-called dative alternations, exemplified by the child-produced sentences 1 to 3, from the Brown (1973) corpus in CHILDES (MacWhinney, 1995).

1. *you took me three scrambled eggs* (a direct object dative (DOD) from Adam at age 3;6)
2. *Mommy can you fix dis for me ?* (a prepositional dative (PD) from Adam at age 4;7)
3. *\*Mommy, fix me my tiger* (from Adam at age 5;2)

Examples like these show that children generalize their use of verbs. For example, in sentence (1), the child Adam uses *take* as a DOD before any recorded occurrence of a similar use of *take* in adult speech to Adam. Such verbs *alternate* because they can also occur with a prepositional form, as in sentence (2). However, sometimes a child’s use of verbs like

these amounts to an overgeneralization – that is, their productive use of a verb in a pattern that does not occur in the adult grammar, as in sentence (3), above. Faced with these two verb frames the task for the learner is to decide for a particular verb if it is a non-alternating DOD only verb, a PD only verb, or an alternating verb that allows both forms.

This ambiguity raises an important learnability question, conventionally known as Baker’s paradox (Baker, 1979). On the assumption that children only receive positive examples of verb forms, then it is not clear how they might recover from the overgeneralization exhibited in sentence (3) above, because they will never receive positive sentences from adults like (3), using *fix* in a DOD form. As has long been noted, if negative examples were systematically available to learners, then this problem would be solved, since the child would be given evidence that the DOD form is not possible in the adult grammar. However, although parental correction could be considered to be a source of negative evidence, it is neither systematic nor generally available to all children (Marcus, 1993). Even when it does occur, all careful studies have indicated that it seems mostly concerned with semantic appropriateness rather than syntax. In the cases where it is related to syntax, it is often difficult to determine what the correction refers to in the utterance and besides children seem to be oblivious to the correction (Brown and Hanlon, 1970; Ingram, 1989).

One alternative solution to Baker’s paradox that has been widely discussed at least since Chomsky (1981) is the use of *indirect negative evidence*. On the indirect negative evidence model, if a verb is not found where it would be expected to occur, the learner may conclude it is not part of the adult grammar. Crucially, the indirect evidence model is inherently statistical. Different formalizations of indirect negative evidence have been incorporated in several computational learning models for learning e.g. grammars (Briscoe, 1997; Villavicencio, 2002; Kwiatkowski et al., 2010); dative verbs (Perfors, Tenenbaum, and Wonnacott, 2010; Hsu and Chater, 2010); and multiword verbs (Nematzadeh, Fazly, and Stevenson, 2013). Since a number of closely related

models can all implement the indirect negative evidence approach, the decision of which one to choose for a given task may not be entirely clear. In this paper we compare a range of statistical models consistent with a certain behavior: early overgeneralization, with eventual convergence to the correct target on the basis of exposure to more data.

### 3 Materials and Methods

#### 3.1 Dative Corpora

To emulate a child language acquisition environment we use naturalistic longitudinal child-directed data, from the Brown corpus in CHILDES, for one child (Adam) for a subset of 19 verbs in the DOD and PD verb frames, figure 1. This dataset was originally reported in Perfors, Tenenbaum, and Wonnacott (2010), and longitudinal and incremental aspects to acquisition are approximated by dividing the data available into 5 incremental epochs (E1 to E5 in the figures), where at the final epoch the learner has seen the full corpus.

Model comparison requires a gold standard database for acquisition, reporting which frames have been learned for which verbs at each stage, and how likely a child is of making creative uses of a particular verb in a new frame. An independent gold standard with developmental information (e.g. Gropen et al. (1989)) would clearly be ideal. Absent this, a first step is demonstrating that simpler alternative models can replicate HBM performance on their own terms. Therefore, the gold standard we use for evaluation is the classification predicted by Perfors, Tenenbaum, and Wonnacott (2010). The evaluations reported in our analysis take into account intrinsic characteristics of each model in relation to the likelihoods of the verbs, to determine the extent to which the models go beyond the data they were exposed to, discussed in section 2. Further, since it has been argued that very low frequency verbs may not yet be firmly placed in a child’s lexicon (Yang, 2010; Gropen et al., 1989), at each epoch we also impose a low-frequency threshold of 5 occurrences, considering only verbs that the learner has seen at least 5 times. This use of a low-frequency threshold for learning has extensive support in the literature for learning

of all kinds in both human and non-human animals, e.g. (Gallistel, 2002). A cut-off frequency in this range has also commonly been used in NLP tasks like POS tagging (Ratnaparkhi, 1999).

### 3.2 The learners

We selected a set of representative statistical models that are capable in principle of solving this classification task, ranging from what is perhaps the simplest possible, a simple binomial, all the way to multi-level hierarchical Bayesian approaches.

A Binomial distribution serves as the simplest model for capturing the behavior of a verb occurring in either DOD or PD frame. Representing the probability of DOD as  $\theta$ , after  $n$  occurrences of the verb the probability that  $y$  of them are DOD is:

$$p(y|\theta, n) = \binom{n}{y} \theta^y (1 - \theta)^{n-y} \quad (1)$$

Considering that  $p(y|\theta, n)$  is the likelihood in a Bayesian framework, the simplest and the most intuitive estimator of  $\theta$ , given  $y$  in  $n$  verb occurrences, is the Maximum Likelihood Estimator (MLE):

$$\theta_{MLE} = \frac{y}{n} \quad (2)$$

$\theta_{MLE}$  is viable as a learning model in the sense that its accuracy increases as the amount of evidence for a verb grows ( $n \rightarrow \infty$ ), reflecting the incremental, on-line character of language learning. However, one well known limitation of MLE is that it assigns zero probability mass to unseen events. Ruling out events on the grounds that they did not occur in a finite data set early in learning may be too strong – though it should be noted that this is simply one (overly strong) version of the indirect negative evidence position.

Again as is familiar, to overcome zero count problem, models adopt one or another method of smoothing to assign a small probability mass to unseen events. In a Bayesian formulation, this amounts to assigning non-zero probability mass to some set of priors; smoothing also captures the notion of *generalization*, making predictions about data that has never been seen by the learner. In the

context of verb learning smoothing could be based on several principles:

- an (innate) expectation as to how verbs in general should behave;
- an acquired class-based expectation of the behavior of a verb, based on its association to similar but more frequent verbs.

The former can be readily implemented in terms of prior probability estimates. As we discuss below, class-based estimates arise from one or another clustering method, and can produce more accurate estimates for less frequent verbs based on patterns already learned for more frequent verbs in the same class; see (Perfors, Tenenbaum, and Wonnacott, 2010). In this case, smoothing is a side-effect of the behavior of a class as a whole.

When learning begins, the prior probability is the only source of information for a learner and, as such, dominates the value of the posterior probability. However, in the large sample limit, it is the likelihood that dominates the posterior distribution regardless of the prior. In Hierarchical Bayesian Models both effects are naturally incorporated. The prior distribution is structured as a chain of distributions of parameters and hyper-parameters, and the data may be divided into classes that share some of the hyper-parameters, as defined below for the case of a three levels model:

$$\begin{aligned} \lambda &\sim \text{Exponential}(1) \\ \mu &\sim \text{Exponential}(1) \\ \alpha_k &\sim \text{Exponential}(\lambda) \\ \beta_k &\sim \text{Beta}(\mu, \mu) \\ \theta_{ik} &\sim \text{Beta}(\alpha_k \beta_k, \alpha_k (1 - \beta_k)) \\ y_i | n_i &\sim \text{Binomial}(\theta_{ik}) \end{aligned}$$

The indices refer to the possible hierarchies among the hyper-parameters.  $\lambda$  and  $\mu$  are in the top, and they are shared by all verbs. Then there are classes of different  $\alpha_k, \beta_k$ , and the probabilities for the DOD frame for the different verbs ( $\theta_{ik}$ ) are drawn according to the classes  $k$  assigned to them. An estimate for ( $\theta_{ik}$ ) for a given configuration of clusters is given by

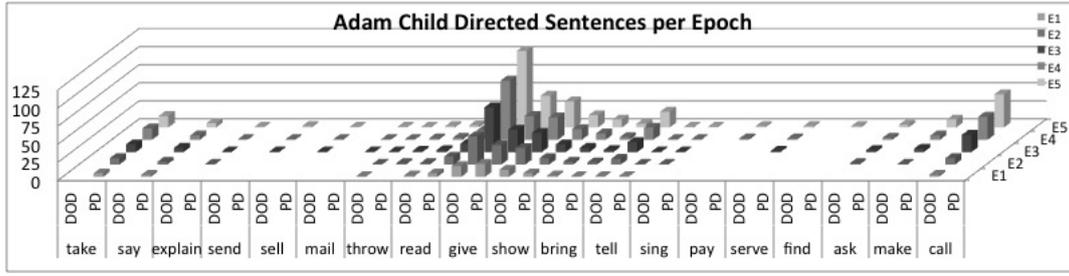


Figure 1: Verb tokens per epoch (E1 to E5)

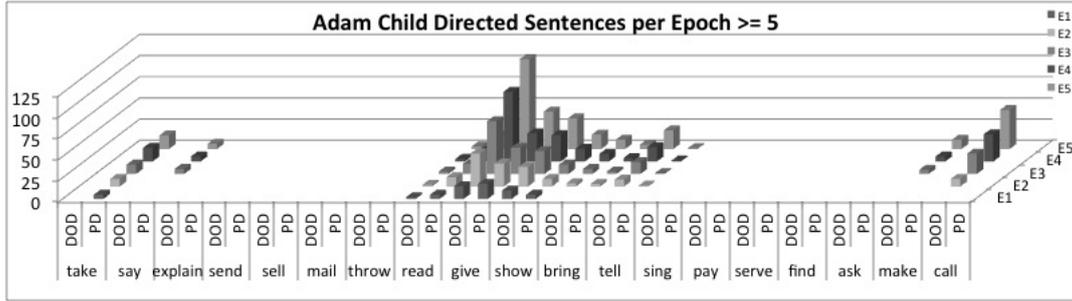


Figure 2: Verb tokens  $\geq 5$  per epoch (E1 to E5)

$$\theta_{HBM}^{ik} = \frac{1}{P(\mathbf{Y})} \int \frac{y_i + \alpha_k \beta_k}{n_i + \alpha_k} P_{L3}^*(\alpha, \beta, \lambda, \mu | \mathbf{Y}) d\alpha d\beta d\lambda d\mu$$

where  $P(\mathbf{Y})$  is the evidence of the data, the unnormalized posterior for the hyper-parameters is

$$P_{L3}^*(\alpha, \beta, \lambda, \mu | \mathbf{Y}) = \left( \prod_k P(\{y_i, n_i\}_{i \in k} | \alpha_k, \beta_k) \text{Exp}(\alpha_k | \lambda) \text{Beta}(\beta_k | \mu, \mu) \right) * \text{Exp}(\lambda | 1) \text{Exp}(\mu | 1)$$

and the likelihood for  $\alpha$  and  $\beta$  is

$$P(\{y_i, n_i\}_{i \in k} | \alpha_k, \beta_k) = \prod_{i \in k} \binom{n_i}{y_i} \frac{B(\alpha_k \beta_k + y_i, \alpha_k (1 - \beta_k) + n_i - y_i)}{B(\alpha_k \beta_k, \alpha_k (1 - \beta_k))}$$

The Hierarchical Bayesian Model prediction for  $\theta_i$  is the average of the estimate  $\theta_{HBM}^{ik}$  over all possible partitions of the verbs in the task. To simplify the notation we can write

$$\theta_{HBM} = E \left[ \frac{y + \alpha \beta}{n + \alpha} \right] \quad (3)$$

where in the expression  $E[\dots]$  are included the integrals described above and the average of all possible class partitions. Due to this complexity, in practice even small data sets require the use of MCMC methods, and statistical models for partitions, like CRP (Gelman et al., 2003; Perfors, Tenenbaum, and Wonnacott,

2010). This complexity also calls into question the cognitive fidelity of such approaches.

Eq.3 is particularly interesting because by fixing  $\alpha$  and  $\beta$  (instead of averaging over them) it is possible to deduce simpler (and classical) models: MLE corresponds to  $\alpha = 0$ ; the so called “add-one” smoothing (referred in this paper as L1) corresponds to  $\alpha = 2$  and  $\beta = 1/2$ . From Eq.3 it is also clear that if  $\alpha$  and  $\beta$  (or their distributions) are unchanged, as the evidence of a verb grows ( $n \rightarrow \infty$ ), the HBM estimate approaches MLE’s, ( $\theta_{HBM} \rightarrow \theta_{MLE}$ ). On the other hand, when  $\alpha \gg n$ ,  $\theta_{HBM} \sim \beta$ , so that  $\beta$  can be interpreted as a prior value for  $\theta$  in the low frequency limit.

Following this reasoning, we propose an alternative approach, a linear competition learner (LCL), that explicitly models the behavior of a given verb as the linear competition between the evidence for the verb, and the average behavior of verbs of the same class. As clustering is defined independently from parameter estimation, the advantages of the proposed approach are twofold. First, it is computationally much simpler, not requiring approximations by Monte Carlo methods. Second, differently from HBMs where the same attributes are used for clustering and parameter estimation (in this case the DOD and PD counts for each verb), in LCL cluster-

ing may be done using more general contexts that employ a variety of linguistic and environmental attributes.

For LCL the prior and class-based information are incorporated as:

$$\theta_{LCL} = \frac{y_i + \alpha_C \beta_C}{n_i + \alpha_C} \quad (4)$$

where  $\alpha_C$  and  $\beta_C$  are defined via justifiable heuristic expressions dependent solely on the statistics of the class attributed to each verb  $i$ .

The strength of the prior ( $\alpha_C$ ) is a monotonic function of the number of elements ( $m_C$ ) in the class  $C$ , excluding the target verb  $v_i$ . To approximate the gold standard behavior of the HBM for this task (Perfors, Tenenbaum, and Wonnacott, 2010) we chose the following function for  $\alpha_C$ :

$$\alpha_C = m_C^{3/2}(1 - m_C^{-1/5}) + 0.1 \quad (5)$$

with the strength of the prior for the LCL model depending on the number of verbs in the class, not on their frequency. Eq.5 was chosen as a good fit to HBMs, without incurring their complexity. The powers are simple fractions, not arbitrary numbers. A best fit was not attempted due to the lack of assessment of how accurate HBMs are on real data.

The prior value ( $\beta_C$ ) is a smoothed estimation of the probability of DOD in a given class, combining the evidence for all verbs in that class:

$$\beta_C = \frac{Y_C + 1/2}{N_C + 1} \quad (6)$$

in this case  $Y_C$  is the number of DOD occurrences in the class, and  $N_C$  the total number of verb occurrences in the class, in both cases excluding the target verb  $v_i$ .

The interpretation of these parameters is as follows:  $\beta_C$  is the estimate of  $\theta$  in the absence of any data for a verb; and  $\alpha_C$  controls the crossover between this estimate and MLE, with a large  $\alpha_C$  requiring a larger sample ( $n_i$ ) to overcome the bias given by  $\beta_C$ .

For comparative purposes, in this paper we examine alternative models for (a) probability estimation and (b) clustering. The models are the following:

- two models without clusters: MLE and L1;

- two models where clusters are performed independently: LCL and MLE $_{\alpha\beta}$ ; and
- the full HBM described before.

MLE $_{\alpha\beta}$  corresponds to replacing  $\alpha, \beta$  in eq.3 by their maximal likelihood values calculated from  $P(\{y_i, n_i\}_{i \in k} | \alpha, \beta)$  described before.

For models without clustering, estimation is based solely on the observed behavior of verbs. With clustering, same-cluster verbs share some parameters, influencing one another. HBMs place distributions over possible clusters, with estimation derived from averages over distributions. In HBMs, clustering and probability estimation are calculated jointly. In the other models these two estimates are calculated separately, permitting 'plug-and-play' use of external clustering methods, like X-means (Pelleg and Moore, 2000)<sup>1</sup>. However, to further assess the impact of cluster assignment on alternative model performance, we also used the clusters that maximize the evidence of the HBM for the DOD and PD counts of the target verbs, and we refer to these as Maximum Evidence (ME) clusters. In MWE clusters, verbs are separated into 3 classes: one if they have counts for both frames; another for only the DOD frame; and a final for only the PD frame.

## 4 Evaluation

The learning task consists of estimating the probability that a given verb occurs in a particular frame, using previous occurrences as the basis for this estimation. In this context, overgeneralization can be viewed as the model's predictions that a given verb seen only in one frame (say, a PD) can also occur in the other (say, a DOD) as well, and it decreases as the learner receives more data. In one extreme we have MLE, which does not overgeneralize, and in the other the L1 model, which assigns uniform probability for all unseen cases. The other 3 models fall somewhere in between, overgeneralizing beyond the observed data, using the prior and class-based smoothing to assign some (low) probability mass to an unseen verb-frame pair. The relevant models'

<sup>1</sup>Other clustering algorithms were also used; here we report X-means results as representative of these models. X-means is available from <http://www.cs.waikato.ac.nz/ml/weka/>

predictions for each of the target verbs in the DOD frame, given the full corpus, are in figure 3. In either end of the figure are the verbs that were attested in only one of the frames (PD only at the left-hand end, and DOD only at the right-hand end). For these verbs, LCL and HBM exhibit similar behavior. When the low-frequency threshold is applied,  $MLE_{\alpha\beta}$ , HBM and LCL work equally well, figure 4.

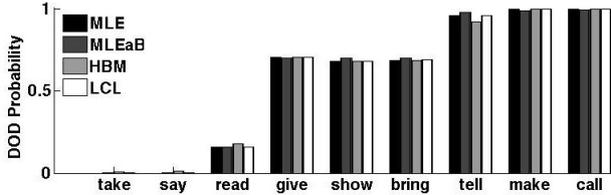


Figure 4: Probability of verbs in DOD frame, Low Frequency Threshold.

To examine how overgeneralization progresses during the course of learning as the models were exposed to increasing amounts of data, we used the corpus divided by cumulative epochs, as described in §3.1. For each epoch, verbs seen in only one of the frames were divided in 5 frequency bins, and the models were assessed as to how much overgeneralization they displayed for each of these verbs. Following Perfors, Tenenbaum, and Wonnacott (2010) overgeneralization is calculated as the absolute difference between the models predicted  $\theta$  and  $\theta_{MLE}$ , for each of the epochs, figure 5, and for comparative purposes their alternating/non-alternating classification is also adopted. For non-alternating verbs, overgeneralization reflects the degree of smoothing of each model. As expected, the more frequent a verb is, the more confident the model is in the indirect negative evidence it has for that verb, and the less it overgeneralizes, shown in the lighter bars in all epochs. In addition, the overall effect of larger amounts of data are indicated by a reduction in overgeneralization epoch by epoch. The effects of class-based smoothing can be assessed comparing L1, a model without clustering which displays a constant degree of overgeneralization regardless of the epoch, while HBM uses a distribution over clusters and the other models X-means. If a low-frequency threshold is applied, the differences between the models

decrease significantly and so does the degree of overgeneralization in the models' predictions, as shown in the 3 lighter bars in the figure.

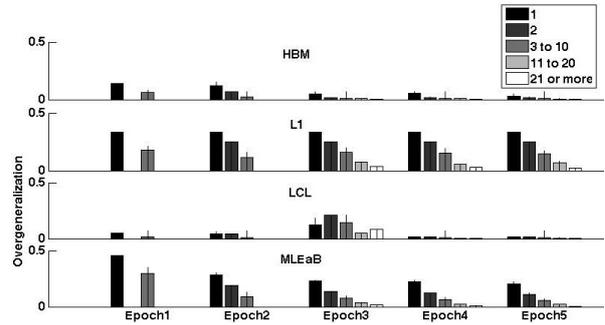


Figure 5: Overgeneralization, per epoch, per frequency bin, where 0.5 corresponds to the maximum overgeneralization.

While the models differ somewhat in their predictions, the quantitative differences need to be assessed more carefully. To compare the models and provide an overall difference measure, we use the predictions of the more complex model, HBM, as a baseline and then calculate the difference between its predictions and those of the other models. We used three different measures for comparing models, one for their standard difference; one that prioritizes agreement for high frequency verbs; and one that focuses more on low frequency verbs.

The first measure, denoted *Difference*, captures a direct comparison between two models,  $M_1$  and  $M_2$  as the average prediction difference among the verbs, and is defined as:

$$D(M_1, M_2) = \frac{1}{N_{verbs}} \sum_{i=1}^{N_{verbs}} abs(\theta_{M_1}^i - \theta_{M_2}^i)$$

This measure treats all differences uniformly, regardless of whether they relate to high or low frequency verbs in the learning sample (e.g. for *bring* with 150 counts and *serve* with only 1 have the same weight). To focus on high frequency verbs, we also define the *Weighted Difference* between two models as:

$$D_n(M_1, M_2) = \frac{1}{\sum_i n_i} \sum_{i=1}^{N_{verbs}} n_i abs(\theta_{M_1}^i - \theta_{M_2}^i)$$

Here we expect  $D_n < D$  since models tend to

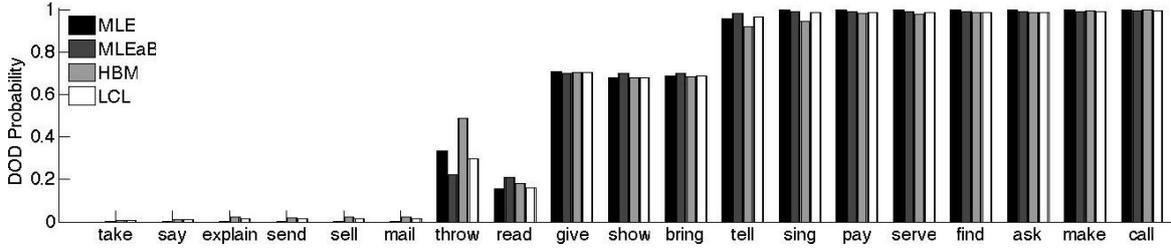


Figure 3: Probability of verbs in DOD frame.

agree as the amount of evidence for each verb increases. Conversely, our third measure, denoted *Inverted*, prioritizes the agreement between two models on low frequency verbs, defined as follows:

$$D_{1/n}(M_1, M_2) = \frac{1}{\sum_i 1/n_i} \sum_{i=1}^{N_{verbs}} \frac{1}{n_i} abs(\theta_{M_1}^i - \theta_{M_2}^i)$$

$D_{1/n}$  captures the degree of similarity in overgeneralization between two models. The results of applying these three difference measures are shown in figure 6 for the relevant models, where grey is for  $D(M_1, M_2)$ , black for  $D_n(M_1, M_2)$  and white for  $D_{1/n}(M_1, M_2)$ . Given the probabilistic nature of Monte Carlo methods, there is also a variation between different runs of the HBM model (HBM to HBM-2), and this indicates that models that perform within these bounds can be considered to be equivalent (e.g. HBMs and ME-MLE $_{\alpha\beta}$  for Weighted Difference, and the HBMs and X-MLE $_{\alpha\beta}$  for the Inverted Difference).

Comparing the prediction agreement, the strong influence of clustering is clear: the models that have HBM compatible clusters have similar performances. For instance, all the models that adopt the ME clusters for the data perform closest to HBMs. Moreover, the weighted differences tend to be smaller than 0.01 and around 0.02 for the inverted differences. The results for these measures become even closer in most cases when the low frequency threshold is adopted, figure 7, as the

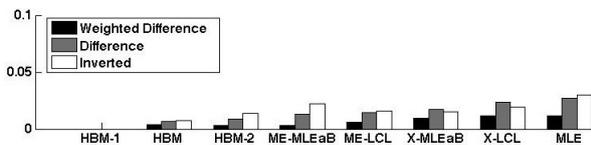


Figure 6: Model Comparisons.

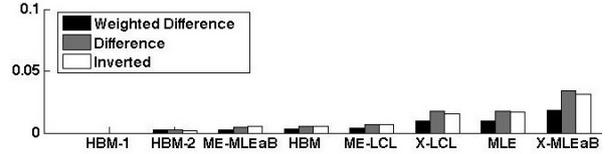


Figure 7: Model Comparison - Low Frequency Threshold.

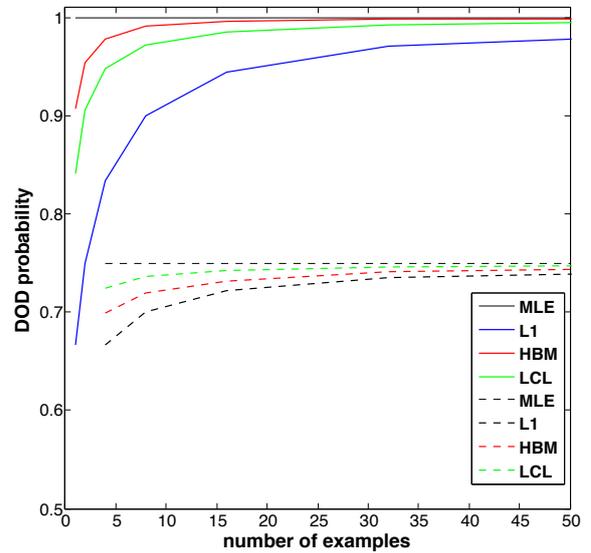


Figure 8: DOD probability evolution for models with increase in evidence

evidence reduces the influence of the prior.

To examine the decay of overgeneralization with the increase in evidence for these models, two simulated scenarios are defined for a single generic verb: one where the evidence for DOD amounts to 75% of the data (dashed lines) and in the other to 100% (solid lines), figures 9 and 8. Unsurprisingly, the performance of the models is dependent on the amount of evidence available. This is a consequence of the decrease in the influence of the priors as the sample size increases in a rate of  $1/N$ , as shown in figure 9 for the decrease in overgeneralization. Ultimately it is the ev-

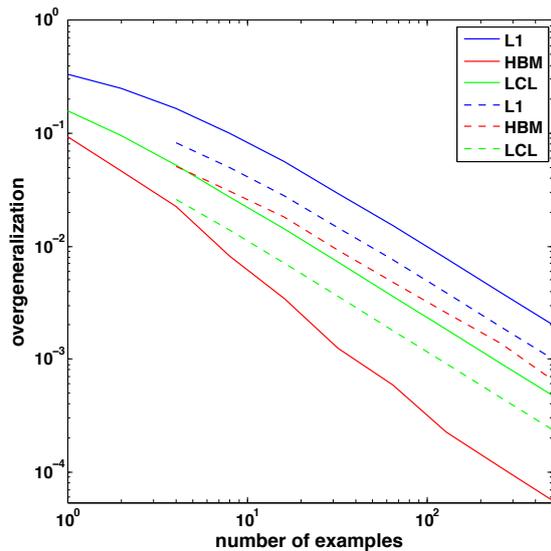


Figure 9: Overgeneralization reduction with increase in evidence

idence that dominates the posterior probability. Although the Bayesian model exhibits fast convergence, after 10 examples, the simpler model L1 is only approximately 3% below the Bayesian model in performance for scenario 1 and is still 90% accurate in scenario 2, figure 8.

These results suggest that while these models all differ slightly in the degree of overgeneralization for low frequency data and noise, these differences are small, and as evidence reaches approximately 10 examples per verb, the overall performance for all models approaches that of MLE.

## 5 Conclusions and Future Work

HBM's have been successfully used for a number of language acquisition tasks capturing both patterns of under- and overgeneralization found in child language acquisition. Their (hyper)parameters provide robustness for dealing with low frequency events, noise, and uncertainty and a good fit to the data, but this fidelity comes at the cost of complex computation. Here we have examined HBM's against computationally simpler approaches to dative alternation acquisition, which implement the indirect negative approach. We also advanced several measures for model comparison in order to quantify their agreement to assist in the task of model selection. The results show that the proposed LCL model, in

particular, that combines class-based smoothing with maximum likelihood estimation, obtains results comparable to those of HBM's, in a much simpler framework. Moreover, when a cognitively-viable frequency threshold is adopted, differences in the performance of all models decrease, and quite rapidly approach the performance of MLE.

In this paper we used standard clustering techniques grounded solely on verb counts to enable comparison with previous work. However, a variety of additional linguistic and distributional features could be used for clustering verbs into more semantically motivated classes, using a larger number of frames and verbs. This will be examined in future work. We also plan to investigate the use of clustering methods more targeted to language tasks (Sun and Korhonen, 2009).

## Acknowledgements

We would like to thank the support of projects CAPES/COFECUB 707/11, CNPq 482520/2012-4, 478222/2011-4, 312184/2012-3, 551964/2011-1 and 312077/2012-2. We also want to thank Amy Perfors for kindly sharing the input data.

## References

- Baker, Carl L. 1979. Syntactic Theory and the Projection Problem. *Linguistic Inquiry*, 10(4):533–581.
- Briscoe, Ted. 1997. Co-evolution of language and the language acquisition device. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 418–427. Morgan Kaufmann.
- Brown, Roger. 1973. *A first language: The early stages*. Harvard University Press, Cambridge, Massachusetts.
- Brown, Roger and Camille Hanlon. 1970. Derivational complexity and the order of acquisition of child's speech. In J. Hays, editor, *Cognition and the Development of Language*. NY: John Wiley.
- Chater, Nick, Joshua B. Tenenbaum, and Alan Yuille. 2006. Probabilistic models of cognition: where next? *Trends in Cognitive Sciences*, 10(7):292 – 293.
- Chomsky, Noam. 1981. *Lectures on government and binding*. Mouton de Gruyter.

- Gallistel, Charles R. 2002. Frequency, contingency, and the information processing theory of conditioning. In P.Sedlmeier and T. Betsch, editors, *Frequency processing and cognition*. Oxford University Press, pages 153–171.
- Gelman, Andrew, John B. Carlin, Hal S. Stern, and Donald B. Rubin. 2003. *Bayesian Data Analysis, Second Edition (Chapman & Hall/CRC Texts in Statistical Science)*. Chapman and Hall/CRC, 2 edition.
- Gropen, Jess, Steve Pinker, Michael Hollander, Richard Goldberg, and Ronald Wilson. 1989. The learnability and acquisition of the dative alternation in English. *Language*, 65(2):203–257.
- Hsu, Anne S. and Nick Chater. 2010. The logical problem of language acquisition: A probabilistic perspective. *Cognitive Science*, 34(6):972–1016.
- Ingram, David. 1989. *First Language Acquisition: Method, Description and Explanation*. Cambridge University Press.
- Jones, Matt and Bradley C. Love. 2011. Bayesian Fundamentalism or Enlightenment? On the explanatory status and theoretical contributions of Bayesian models of cognition. *Behavioral and Brain Sciences*, 34(04):169–188.
- Kwiatkowski, Tom, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1223–1233.
- Kwisthout, Johan, Todd Wareham, and Iris van Rooij. 2011. Bayesian intractability is not an ailment that approximation can cure. *Cognitive Science*, 35(5):779–1007.
- Levin, B. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press, Chicago, IL.
- MacWhinney, Brian. 1995. *The CHILDES project: tools for analyzing talk*. Hillsdale, NJ: Lawrence Erlbaum Associates, second edition.
- Marcus, Gary F. 1993. Negative evidence in language acquisition. *Cognition*, 46:53–85.
- Marr, D. 1982. *Vision*. San Francisco, CA: W. H. Freeman.
- Nematzadeh, Aida, Afsaneh Fazly, and Suzanne Stevenson. 2013. Child acquisition of multiword verbs: A computational investigation. In A. Villavicencio, T. Poibeau, A. Korhonen, and A. Alishahi, editors, *Cognitive Aspects of Computational Language Acquisition*. Springer, pages 235–256.
- Parisien, Christopher, Afsaneh Fazly, and Suzanne Stevenson. 2008. An incremental bayesian model for learning syntactic categories. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning, CoNLL '08*, pages 89–96, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Parisien, Christopher and Suzanne Stevenson. 2010. Learning verb alternations in a usage-based bayesian model. In *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*.
- Pelleg, Dan and Andrew Moore. 2000. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734, San Francisco. Morgan Kaufmann.
- Perfors, Amy, Joshua B. Tenenbaum, and Elizabeth Wonnacott. 2010. Variability, negative evidence, and the acquisition of verb argument constructions. *Journal of Child Language*, (37):607–642.
- Ratnaparkhi, Adwait. 1999. Learning to parse natural language with maximum entropy models. *Machine Learning*, pages 151–175.
- Shalizi, Cosma R. 2009. Dynamics of bayesian updating with dependent data and misspecified models. *ElectroCosmanic Journal of Statistics*, 3:1039–1074.
- Sun, Lin and Anna Korhonen. 2009. Improving verb clustering with automatically acquired selectional preferences. In *EMNLP*, pages 638–647.
- Villavicencio, Aline. 2002. *The Acquisition of a Unification-Based Generalised Categorical Grammar*. Ph.D. thesis, Computer Laboratory, University of Cambridge.
- Wonnacott, Elizabeth, Elissa L. Newport, and Michael K. Tanenhaus. 2008. Acquiring and processing verb argument structure: Distributional learning in a miniature language. *Cognitive Psychology*, 56:165–209.
- Yang, Charles. 2010. Three factors in language variation. *Lingua*, 120:1160–1177.

# A Two Level Model for Context Sensitive Inference Rules

Oren Melamud<sup>§</sup>, Jonathan Berant<sup>†</sup>, Ido Dagan<sup>§</sup>, Jacob Goldberger<sup>◇</sup>, Idan Szpektor<sup>‡</sup>

<sup>§</sup> Computer Science Department, Bar-Ilan University

<sup>†</sup> Computer Science Department, Stanford University

<sup>◇</sup> Faculty of Engineering, Bar-Ilan University

<sup>‡</sup> Yahoo! Research Israel

{melamuo, dagan, goldbej}@{cs, cs, eng}.biu.ac.il

joberant@stanford.edu

idan@yahoo-inc.com

## Abstract

Automatic acquisition of inference rules for predicates has been commonly addressed by computing distributional similarity between vectors of argument words, operating at the word space level. A recent line of work, which addresses context sensitivity of rules, represented contexts in a latent topic space and computed similarity over topic vectors. We propose a novel two-level model, which computes similarities between word-level vectors that are biased by topic-level context representations. Evaluations on a naturally-distributed dataset show that our model significantly outperforms prior word-level and topic-level models. We also release a first context-sensitive inference rule set.

## 1 Introduction

Inference rules for predicates have been identified as an important component in semantic applications, such as Question Answering (QA) (Ravichandran and Hovy, 2002) and Information Extraction (IE) (Shinyama and Sekine, 2006). For example, the inference rule ‘ $X \text{ treat } Y \rightarrow X \text{ relieve } Y$ ’ can be useful to extract pairs of drugs and the illnesses which they relieve, or to answer a question like “Which drugs relieve headache?”. Along this vein, such inference rules constitute a crucial component in generic modeling of textual inference, under the Textual Entailment paradigm (Dagan et al., 2006; Dinu and Wang, 2009).

Motivated by these needs, substantial research was devoted to automatic learning of inference rules from corpora, mostly in an unsupervised distributional setting. This research line was mainly initiated by the highly-cited DIRT algorithm (Lin and Pantel, 2001), which learns inference for binary predicates with two argument slots (like the

rule in the example above). DIRT represents a predicate by two vectors, one for each of the argument slots, where the vector entries correspond to the argument words that occurred with the predicate in the corpus. Inference rules between pairs of predicates are then identified by measuring the similarity between their corresponding argument vectors. This general scheme was further enhanced in several directions, e.g. directional similarity (Bhagat et al., 2007; Szpektor and Dagan, 2008) and meta-classification over similarity values (Berant et al., 2011). Consequently, several knowledge resources of inference rules were released, containing the top scoring rules for each predicate (Schoenmackers et al., 2010; Berant et al., 2011; Nakashole et al., 2012).

The above mentioned methods provide a single confidence score for each rule, which is based on the obtained degree of argument-vector similarities. Thus, a system that applies an inference rule to a text may estimate the validity of the rule application based on the pre-specified rule score. However, the validity of an inference rule may depend on the context in which it is applied, such as the context specified by the given predicate’s arguments. For example, ‘ $AT\&T \text{ acquire } T\text{-Mobile} \rightarrow AT\&T \text{ purchase } T\text{-Mobile}$ ’, is a valid application of the rule ‘ $X \text{ acquire } Y \rightarrow X \text{ purchase } Y$ ’, while ‘ $Children \text{ acquire } skills \rightarrow Children \text{ purchase } skills$ ’ is not. To address this issue, a line of works emerged which computes a *context-sensitive* reliability score for each rule *application*, based on the given context.

The major trend in context-sensitive inference models utilizes latent or class-based methods for context modeling (Pantel et al., 2007; Szpektor et al., 2008; Ritter et al., 2010; Dinu and Lapata, 2010b). In particular, the more recent methods (Ritter et al., 2010; Dinu and Lapata, 2010b) modeled predicates in context as a probability distribution over topics learned by a Latent Dirichlet Allo-

cation (LDA) model. Then, similarity is measured between the two topic distribution vectors corresponding to the two sides of the rule in the given context, yielding a context-sensitive score for each particular rule application.

We notice at this point that while context-insensitive methods represent predicates by argument vectors in the original fine-grained word space, context-sensitive methods represent them as vectors at the level of latent topics. This raises the question of whether such coarse-grained topic vectors might be less informative in determining the semantic similarity between the two predicates.

To address this hypothesized caveat of prior context-sensitive rule scoring methods, we propose a novel generic scheme that integrates word-level and topic-level representations. Our scheme can be applied on top of any context-insensitive “base” similarity measure for rule learning, which operates at the word level, such as Cosine or Lin (Lin, 1998). Rather than computing a single context-insensitive rule score, we compute a distinct word-level similarity score for each topic in an LDA model. Then, when applying a rule in a given context, these different scores are weighed together based on the specific topic distribution under the given context. This way, we calculate similarity over vectors in the original word space, while biasing them towards the given context via a topic model.

In order to promote replicability and equal-term comparison with our results, we based our experiments on publicly available datasets, both for unsupervised learning of the evaluated models and for testing them over a random sample of rule applications. We apply our two-level scheme over three state-of-the-art context-insensitive similarity measures. The evaluation compares performances both with the original context-insensitive measures and with recent LDA-based context-sensitive methods, showing consistent and robust advantages of our scheme. Finally, we release a context-sensitive rule resource comprising over 2,000 frequent verbs and one million rules.

## 2 Background and Model Setting

This section presents components of prior work which are included in our model and experiments, setting the technical preliminaries for the rest of the paper. We first present context-insensitive rule

learning, based on distributional similarity at the word level, and then context-sensitive scoring for rule applications, based on topic-level similarity. Some further discussion of related work appears in Section 6.

### 2.1 Context-insensitive Rule Learning

A predicate inference rule ‘ $LHS \rightarrow RHS$ ’, such as ‘ $X \text{ acquire } Y \rightarrow X \text{ purchase } Y$ ’, specifies a directional inference relation between two predicates. Each rule side consists of a lexical predicate and (two) variable slots for its arguments.<sup>1</sup> Different representations have been used to specify predicates and their argument slots, such as word lemma sequences, regular expressions and dependency parse fragments. A rule can be *applied* when its LHS matches a predicate with a pair of arguments in a text, allowing us to infer its RHS, with the corresponding instantiations for the argument variables. For example, given the text “*AT&T acquires T-Mobile*”, the above rule infers “*AT&T purchases T-Mobile*”.

The DIRT algorithm (Lin and Pantel, 2001) follows the distributional similarity paradigm to learn predicate inference rules. For each predicate, DIRT represents each of its argument slots by an *argument vector*. We denote the two vectors of the  $X$  and  $Y$  slots of a predicate  $pred$  by  $v_{pred}^x$  and  $v_{pred}^y$ , respectively. Each entry of a vector  $v$  corresponds to a particular word (or term)  $w$  that instantiated the argument slot in a learning corpus, with a value  $v(w) = PMI(pred, w)$  (with PMI standing for point-wise mutual information).

To learn inference rules, DIRT considers (in principle) each pair of binary predicates that occurred in the corpus for a candidate rule, ‘ $LHS \rightarrow RHS$ ’. Then, DIRT computes a *reliability score* for the rule by combining the measured similarities between the corresponding argument vectors of the two rule sides. Concretely, denoting by  $l$  and  $r$  the predicates appearing in the two rule sides, DIRT’s reliability score is defined as follows:

$$\begin{aligned} \text{score}_{\text{DIRT}}(LHS \rightarrow RHS) \\ = \sqrt{\text{sim}(v_l^x, v_r^x) \cdot \text{sim}(v_l^y, v_r^y)} \end{aligned} \quad (1)$$

where  $\text{sim}(v, v')$  is a vector similarity measure. Specifically, DIRT employs the Lin similarity

<sup>1</sup>We follow most of the inference-rule learning literature, which focused on binary predicates. However, our context-sensitive scheme can be applied to any arity.

measure from (Lin, 1998), defined as follows:

$$Lin(v, v') = \frac{\sum_{w \in v \cap v'} [v(w) + v'(w)]}{\sum_{w \in v \cup v'} [v(w) + v'(w)]} \quad (2)$$

We note that the general DIRT scheme may be used while employing other “base” vector similarity measures. For example, the *Lin* measure is symmetric, and thus using it would yield the same reliability score when swapping the two sides of a rule. This issue has been addressed in a separate line of research which introduced directional similarity measures suitable for inference relations (Bhagat et al., 2007; Szpektor and Dagan, 2008; Kotlerman et al., 2010). In our experiments we apply our proposed context-sensitive similarity scheme over three different base similarity measures.

DIRT and similar context-insensitive inference methods provide a single reliability score for a learned inference rule, which aims to predict the validity of the rule’s applications. However, as exemplified in the Introduction, an inference rule may be valid in some contexts but invalid in others (e.g. *acquiring* entails *purchasing* for *goods*, but not for *skills*). Since vector similarity in DIRT is computed over the single aggregate argument vector, the obtained reliability score tends to be biased towards the dominant contexts of the involved predicates. For example, we may expect a higher score for ‘*acquire* → *purchase*’ than for ‘*acquire* → *learn*’, since the former matches a more frequent sense of *acquire* in a typical corpus. Following this observation, it is desired to obtain a *context-sensitive reliability score* for each rule application in a given context, as described next.

## 2.2 Context-sensitive Rule Applications

To assess the reliability of applying an inference rule in a given context we need some model for context representation, that should affect the rule reliability score. A major trend in past work is to represent contexts in a reduced-dimensionality latent or class-based model. A couple of earlier works utilized a cluster-based model (Pantel et al., 2007) and an LSA-based model (Szpektor et al., 2008), in a selectional-preferences style approach. Several more recent works utilize a Latent Dirichlet Allocation (LDA) (Blei et al., 2003) framework. We now present an underlying unified view of the *topic-level* models in (Ritter et al., 2010; Dinu and Lapata, 2010b), which we follow in our

own model and in comparative model evaluations. We note that a similar LDA model construction was employed also in (Séaghdha, 2010), for estimating predicate-argument likelihood.

First, an LDA model is constructed, as follows. Similar to the construction of argument vectors in the distributional model (described above in subsection 2.1), all arguments instantiating each predicate slot are extracted from a large learning corpus. Then, for each slot of each predicate, a pseudo-document is constructed containing the set of all argument words that instantiated this slot in the corpus. We denote the two documents constructed for the *X* and *Y* slots of a predicate *pred* by  $d_{pred}^x$  and  $d_{pred}^y$ , respectively. In comparison to the distributional model, these two documents correspond to the analogous argument vectors  $v_{pred}^x$  and  $v_{pred}^y$ , both containing exactly the same set of words.

Next, an LDA model is learned from the set of all pseudo-documents, extracted for all predicates.<sup>2</sup> The learning process results in the construction of *K* latent topics, where each topic *t* specifies a distribution over all words, denoted by  $p(w|t)$ , and a topic distribution for each pseudo-document *d*, denoted by  $p(t|d)$ .

Within the LDA model we can derive the a-posteriori topic distribution conditioned on a particular word within a document, denoted by  $p(t|d, w) \propto p(w|t) \cdot p(t|d)$ . In the topic-level model, *d* corresponds to a predicate slot and *w* to a particular argument word instantiating this slot. Hence,  $p(t|d, w)$  is viewed as specifying the relevance (or likelihood) of the topic *t* for the predicate slot in the context of the given argument instantiation. For example, for the predicate slot ‘*acquire Y*’ in the context of the argument ‘*IBM*’, we expect high relevance for a topic about companies, while in the context of the argument ‘*knowledge*’ we expect high relevance for a topic about abstract concepts. Accordingly, the distribution  $p(t|d, w)$  over all topics provides a topic-level representation for a predicate slot in the context of a particular argument *w*. This representation is used by the topic-level model to compute a context-sensitive score for inference rule applications, as follows.

<sup>2</sup>We note that there are variants in the type of LDA model and the way the pseudo-documents are constructed in the referenced prior work. In order to focus on the inference methods rather than on the underlying LDA model, we use the LDA framework described in this paper for all compared methods.

Consider the application of an inference rule ‘ $LHS \rightarrow RHS$ ’ in the context of a particular pair of arguments for the  $X$  and  $Y$  slots, denoted by  $w_x$  and  $w_y$ , respectively. Denoting by  $l$  and  $r$  the predicates appearing in the two rule sides, the reliability score of the topic-level model is defined as follows (we present a geometric mean formulation for consistency with DIRT):

$$\begin{aligned} \text{score}_{\text{Topic}}(LHS \rightarrow RHS, w_x, w_y) \\ = \sqrt{\text{sim}(d_l^x, d_r^x, w_x) \cdot \text{sim}(d_l^y, d_r^y, w_y)} \end{aligned} \quad (3)$$

where  $\text{sim}(d, d', w)$  is a topic-distribution similarity measure conditioned on a given context word. Specifically, Ritter et al. (2010) utilized the dot product form for their similarity measure:

$$\text{sim}_{\text{DC}}(d, d', w) = \Sigma_t[p(t|d, w) \cdot p(t|d', w)] \quad (4)$$

(the subscript DC stands for double-conditioning, as both distributions are conditioned on the argument word, unlike the measure below).

Dinu and Lapata (2010b) presented a slightly different similarity measure for topic distributions that performed better in their setting as well as in a related later paper on context-sensitive scoring of lexical similarity (Dinu and Lapata, 2010a). In this measure, the topic distribution for the right hand side of the rule is not conditioned on  $w$ :

$$\text{sim}_{\text{SC}}(d, d', w) = \Sigma_t[p(t|d, w) \cdot p(t|d')] \quad (5)$$

(the subscript SC stands for single-conditioning, as only the left distribution is conditioned on the argument word). They also experimented with a few variants for the structure of the similarity measure and assessed that best results are obtained with the dot product form. In our experiments, we employ these two similarity measures for topic distributions as baselines representing topic-level models.

Comparing the context-insensitive and context-sensitive models, we see that both of them measure similarity between vector representations of corresponding predicate slots. However, while DIRT computes  $\text{sim}(v, v')$  over vectors in the original word-level space, topic-level models compute  $\text{sim}(d, d', w)$  by measuring similarity of vectors in a reduced-dimensionality latent space. As conjectured in the introduction, such coarse-grain representation might lead to loss of information. Hence, in the next section we propose a combined two-level model, which represents predicate

slots in the original word-level space while biasing the similarity measure through topic-level context models.

### 3 Two-level Context-sensitive Inference

Our model follows the general DIRT scheme while extending it to handle context-sensitive scoring of rule applications, addressing the scenario dealt by the context-sensitive topic models. In particular, we define the context-sensitive score  $\text{score}_{\text{WT}}$ , where  $WT$  stands for the combination of the Word/Topic levels:

$$\begin{aligned} \text{score}_{\text{WT}}(LHS \rightarrow RHS, w_x, w_y) \\ = \sqrt{\text{sim}(v_l^x, v_r^x, w_x) \cdot \text{sim}(v_l^y, v_r^y, w_y)} \end{aligned} \quad (6)$$

Thus, our model computes similarity over word-level (rather than topic-level) argument vectors, while biasing it according to the specific argument words in the given rule application context. The core of our contribution is thus defining the context-sensitive word-level vector similarity measure  $\text{sim}(v, v', w)$ , as described in the remainder of this section.

Following the methods in Section 2, for each predicate  $pred$  we construct, from the learning corpus, its argument vectors  $v_{pred}^x$  and  $v_{pred}^y$  as well as its argument pseudo-documents  $d_{pred}^x$  and  $d_{pred}^y$ . For convenience, when referring to an argument vector  $v$ , we will denote the corresponding pseudo-document by  $d_v$ . Based on all pseudo-documents we learn an LDA model and obtain its associated probability distributions.

The calculation of  $\text{sim}(v, v', w)$  is composed of two steps. At learning time, we compute for each candidate rule a separate, topic-biased, similarity score per each of the topics in the LDA model. Then, at rule application time, we compute an overall reliability score for the rule by combining the per-topic similarity scores, while biasing the score combination according to the given context of  $w$ . These two steps are described in the following two subsections.

#### 3.1 Topic-biased Word-vector Similarities

Given a pair of word vectors  $v$  and  $v'$ , and any desired ‘‘base’’ vector similarity measure  $\text{sim}$  (e.g.  $\text{sim}_{\text{Lin}}$ ), we compute a *topic-biased* similarity score for each LDA topic  $t$ , denoted by  $\text{sim}_t(v, v')$ .  $\text{sim}_t(v, v')$  is computed by applying

the original similarity measure over topic-biased versions of  $v$  and  $v'$ , denoted by  $v_t$  and  $v'_t$ :

$$\text{sim}_t(v, v') = \text{sim}(v_t, v'_t)$$

where

$$v_t(w) = v(w) \cdot p(t|d_v, w)$$

That is, each value in the biased vector,  $v_t(w)$ , is obtained by weighing the original value  $v(w)$  by the relevance of the topic  $t$  to the argument word  $w$  within  $d_v$ . This way, rather than replacing altogether the word-level values  $v(w)$  by the topic probabilities  $p(t|d_v, w)$ , as done in the topic-level models, we use the latter to only bias the former while preserving fine-grained word-level representations. The notation  $\text{Lin}_t$  denotes the  $\text{sim}_t$  measure when applied using  $\text{Lin}$  as the base similarity measure  $\text{sim}$ .

This learning process results in  $K$  different topic-biased similarity scores for each candidate rule, where  $K$  is the number of LDA topics. Table 1 illustrates topic-biased similarities for the  $Y$  slot of two rules involving the predicate ‘*acquire*’. As can be seen, the topic-biased score  $\text{Lin}_t$  for ‘*acquire*  $\rightarrow$  *learn*’ for  $t_2$  is higher than the  $\text{Lin}$  score, since this topic is characterized by arguments that commonly appear with both predicates of the rule. Consequently, the two predicates are found to be distributionally similar when biased for this topic. On the other hand, the topic-biased similarity for  $t_1$  is substantially lower, since prominent words in this topic are likely to occur with ‘*acquire*’ but not with ‘*learn*’, yielding low distributional similarity. Opposite behavior is exhibited for the rule ‘*acquire*  $\rightarrow$  *purchase*’.

### 3.2 Context-sensitive Similarity

When applying an inference rule, we compute for each slot its context-sensitive similarity score  $\text{sim}_{\text{WT}}(v, v', w)$ , where  $v$  and  $v'$  are the slot’s argument vectors for the two rule sides and  $w$  is the word instantiating the slot in the given rule application. This score is computed as a weighted average of the rule’s  $K$  topic-biased similarity scores  $\text{sim}_t$ . In this average, each topic is weighed by its “relevance” for the context in which the rule is applied, which consists of the left-hand-side predicate  $v$  and the argument  $w$ . This relevance is cap-

Topic	$t_1$	$t_2$
Top 5 words	calbiochem	rights
	corel	syndrome
	networks	majority
	viacom	knowledge
	financially	skill
<i>acquire</i> $\rightarrow$ <i>learn</i>		
$\text{Lin}_t(v, v')$	0.040	0.334
$\text{Lin}(v, v')$	0.165	
<i>acquire</i> $\rightarrow$ <i>purchase</i>		
$\text{Lin}_t(v, v')$	0.427	0.241
$\text{Lin}(v, v')$	0.267	

Table 1: Two characteristic topics for the  $Y$  slot of ‘*acquire*’, along with their topic-biased  $\text{Lin}$  similarities scores  $\text{Lin}_t$ , compared with the original  $\text{Lin}$  similarity, for two rules. The relevance of each topic to different arguments of ‘*acquire*’ is illustrated by showing the top 5 words in the argument vector  $v_{\text{acquire}}^y$  for which the illustrated topic is the most likely one.

tured by  $p(t|d_v, w)$ :

$$\text{sim}_{\text{WT}}(v, v', w) = \sum_t [p(t|d_v, w) \cdot \text{sim}_t(v, v')] \quad (7)$$

This way, a rule application would obtain a high score only if the current context fits those topics for which the rule is indeed likely to be valid, as captured by a high topic-biased similarity. The notation  $\text{Lin}_{\text{WT}}$  denotes the  $\text{sim}_{\text{WT}}$  measure, when using  $\text{Lin}_t$  as the topic-biased similarity measure.

Table 2 illustrates the calculation of context-sensitive similarity scores in four rule applications, involving the  $Y$  slot of the predicate ‘*acquire*’. We observe that relative to the fixed context-insensitive  $\text{Lin}$  score, the score of ‘*acquire*  $\rightarrow$  *learn*’ is substantially promoted for the argument ‘*skill*’ while being demoted for ‘*Skype*’. The opposite behavior is observed for ‘*acquire*  $\rightarrow$  *purchase*’, altogether demonstrating how our model successfully biases the similarity score according to rule validity in context.

## 4 Experimental Settings

To evaluate our model, we compare it both to context-insensitive similarity measures as well as to prior context-sensitive methods. Furthermore, to better understand its applicability in typical NLP tasks, we focus on an evaluation setting that corresponds to a natural distribution of examples from a large corpus.

Topic	$t_1$	$t_2$
Top 5 words	calbiochem corel networks viacom financially	rights syndrome majority knowledge skill
'acquire Skype $\rightarrow$ learn Skype'		
$p(t d_v, w)$	0.974	0.000
$\text{Lin}_t(v, v')$	0.040	0.334
$\text{Lin}_{WT}(v, v', w)$	<b>0.039</b>	
$\text{Lin}(v, v')$	0.165	
'acquire Skype $\rightarrow$ purchase Skype'		
$p(t d_v, w)$	0.974	0.000
$\text{Lin}_t(v, v')$	0.427	0.241
$\text{Lin}_{WT}(v, v', w)$	<b>0.417</b>	
$\text{Lin}(v, v')$	0.267	
'acquire skill $\rightarrow$ learn skill'		
$p(t d_v, w)$	0.000	0.380
$\text{Lin}_t(v, v')$	0.040	0.334
$\text{Lin}_{WT}(v, v', w)$	<b>0.251</b>	
$\text{Lin}(v, v')$	0.165	
'acquire skill $\rightarrow$ purchase skill'		
$p(t d_v, w)$	0.000	0.380
$\text{Lin}_t(v, v')$	0.427	0.241
$\text{Lin}_{WT}(v, v', w)$	<b>0.181</b>	
$\text{Lin}(v, v')$	0.267	

Table 2: Context-sensitive similarity scores (in bold) for the  $Y$  slots of four rule applications. The components of the score calculation are shown for the topics of Table 1. For each rule application, the table shows a couple of the topic-biased scores  $\text{Lin}_t$  of the rule (as in Table 1), along with the topic relevance for the given context  $p(t|d_v, w)$ , which weighs the topic-biased scores in the  $\text{Lin}_{WT}$  calculation. The context-insensitive  $\text{Lin}$  score is shown for comparison.

#### 4.1 Evaluated Rule Application Methods

We evaluated the following rule application methods: the original context-insensitive word model, following DIRT (Lin and Pantel, 2001), as described in Equation 1, denoted by CI; our own topic-word context-sensitive model, as described in Equation 6, denoted by WT. In addition, we evaluated two variants of the topic-level context-sensitive model, denoted DC and SC. DC follows the double conditioned contextualized similarity measure according to Equation 4, as implemented by (Ritter et al., 2010), while SC follows the single conditioned one at Equation 5, as implemented by (Dinu and Lapata, 2010b; Dinu and Lapata, 2010a).

Since our model can contextualize various distributional similarity measures, we evaluated the performance of all the above methods on several base similarity measures and their learned rule-

sets, namely Lin (Lin, 1998), BInc (Szpektor and Dagan, 2008) and vector Cosine similarity. The Lin similarity measure is described in Equation 2. Binc (Szpektor and Dagan, 2008) is a directional similarity measure between word vectors, which outperformed Lin for predicate inference (Szpektor and Dagan, 2008).

To build the rule-sets and models for the tested approaches we utilized the ReVerb corpus (Fader et al., 2011), a large scale publicly available web-based open extractions data set, containing about 15 million unique template extractions.<sup>3</sup> ReVerb template extractions/instantiations are in the form of a tuple  $(x, pred, y)$ , containing  $pred$ , a verb predicate,  $x$ , the argument instantiation of the template’s slot  $X$ , and  $y$ , the instantiation of the template’s slot  $Y$ .

ReVerb includes over 600,000 different templates that comprise a verb but may also include other words, for example ‘ $X$  can accommodate up to  $Y$ ’. Yet, many of these templates share a similar meaning, e.g. ‘ $X$  accommodate up to  $Y$ ’, ‘ $X$  can accommodate up to  $Y$ ’, ‘ $X$  will accommodate up to  $Y$ ’, etc. Following Sekine (2005), we clustered templates that share their main verb predicate in order to scale down the number of different predicates in the corpus and collect richer word co-occurrence statistics per predicate.

Next, we applied some clean-up preprocessing to the ReVerb extractions. This includes discarding stop words, rare words and non-alphabetical words instantiating either the  $X$  or the  $Y$  arguments. In addition, we discarded all predicates that co-occur with less than 100 unique argument words in each slot. The remaining corpus consists of 7 million unique extractions and 2,155 verb predicates.

Finally, we trained an LDA model, as described in Section 2, using Mallet (McCallum, 2002). Then, for each original context-insensitive similarity measure, we learned from ReVerb a rule-set comprised of the top 500 rules for every identified predicate. To complete the learning, we calculated the topic-biased similarity score for each learned rule under each LDA topic, as specified in our context-sensitive model. We release a rule set comprising the top 500 context-sensitive rules that we learned for each of the verb predicates in our learning corpus, along with our trained LDA

<sup>3</sup>ReVerb is available at <http://reverb.cs.washington.edu/>

Method	Lin	BInc	Cosine
Valid	266	254	272
Invalid	545	523	539
Total	811	777	811

Table 3: Sizes of rule application test set for each learned rule-set.

model.<sup>4</sup>

## 4.2 Evaluation Task

To evaluate the performance of the different methods we chose the dataset constructed by Zeichner et al. (2012).<sup>5</sup> This publicly available dataset contains about 6,500 manually annotated predicate template rule applications, each one labeled as correct or incorrect. For example, ‘*Jack agree with Jill*  $\rightarrow$  *Jack feel sorry for Jill*’ is a rule application in this dataset, labeled as incorrect, and ‘*Registration open this month*  $\rightarrow$  *Registration begin this month*’ is another rule application, labeled as correct. Rule applications were generated by randomly sampling extractions from ReVerb, such as (‘*Jack*’, ‘*agree with*’, ‘*Jill*’) and then sampling possible rules for each, such as ‘*agree with*  $\rightarrow$  *feel sorry for*’. Hence, this dataset provides naturally distributed rule inferences with respect to ReVerb.

Whenever we evaluated a distributional similarity measure (namely Lin, BInc, or Cosine), we discarded instances from Zeichner et al.’s dataset in which the assessed rule is not in the context-insensitive rule-set learned for this measure or the argument instantiation of the rule is not in the LDA lexicon. We refer to the remaining instances as the *test set* per measure, e.g. Lin’s test set. Table 3 details the size of each such test set in our experiment.

Finally, the task under which we assessed the tested models is to rank all rule applications in each test set, aiming to rank the valid rule applications above the invalid ones.

## 5 Results

We evaluated the performance of each tested method by measuring Mean Average Precision (MAP) (Manning et al., 2008) of the rule application ranking computed by this method. In order

<sup>4</sup>Our resource is available at: <http://www.cs.biu.ac.il/~nlp/downloads/wt-rules.html>

<sup>5</sup>The dataset is available at: <http://www.cs.biu.ac.il/~nlp/downloads/annotation-rule-application.htm>

Method	Lin	BInc	Cosine
CI	0.503	0.513	0.513
DC	0.451 (1200)	0.455 (1200)	0.455 (1200)
SC	0.443 (1200)	0.458 (1200)	0.452 (1200)
WT	<b>0.562</b> (100)	<b>0.584</b> (50)	<b>0.565</b> (25)

Table 4: MAP values on corresponding test set obtained by each method. Figures in parentheses indicate optimal number of LDA topics.

to compute MAP values and corresponding statistical significance, we randomly split each test set into 30 subsets. For each method we computed Average Precision on every subset and then took the average over all subsets as the MAP value.

Since all tested context-sensitive approaches are based on LDA topics, we varied for each method the number of LDA topics  $K$  that optimizes its performance, ranging from 25 to 1600 topics. We used LDA hyperparameters  $\beta = 0.01$  and  $\alpha = 0.1$  for  $K < 600$  and  $\alpha = \frac{50}{K}$  for  $K \geq 600$ .

Table 4 presents the optimal MAP performance of each tested measure. Our main result is that our model outperforms all other methods, both context-insensitive and context-sensitive, by a relative increase of more than 10% for all three similarity measures that we tested. This improvement is statistically significant at  $p < 0.01$  for BInc and Lin, and  $p < 0.015$  for Cosine, using paired t-test. This shows that our model indeed successfully leverages contextual information beyond the basic context-agnostic rule scores and is robust across measures.

Surprisingly, both baseline topic-level context-sensitive methods, namely DC and SC, underperformed compared to their context-insensitive baselines. While Dinu and Lapata (Dinu and Lapata, 2010b) did show improvement over context-insensitive DIRT, this result was obtained on the verbs of the Lexical Substitution Task in SemEval (McCarthy and Navigli, 2007), which was manually created with a bias for context-sensitive substitutions. However, our result suggests that topic-level models might not be robust enough when applied to a random sample of inferences.

An interesting indication of the differences between our word-topic model, WT, and topic-only models, DC and SC, lies in the optimal number of LDA topics required for each method. The number of topics in the range 25-100 performed almost equally well under the WT model for all base measures, with a moderate decline for higher numbers.

The need for this rather small number of topics is due to the nature of utilization of topics in WT. Specifically, topics are leveraged for high-level domain disambiguation, while fine grained word-level distributional similarity is computed for each rule under each such domain. This works best for a relatively low number of topics. However, in higher numbers, topics relate to narrower domains and then topic biased word level similarity may become less effective due to potential sparseness. On the other hand, DC and SC rely on topics as a surrogate to predicate-argument co-occurrence features, and thus require a relatively large number of them to be effective.

Delving deeper into our test-set, Zeichner et al. provided a more detailed annotation for each invalid rule application. Specifically, they annotated whether the context under which the rule is applied is valid. For example, in ‘*John bought my car*  $\rightarrow$  *John sold my car*’ the inference is invalid due to an inherently incorrect rule, but the context is valid. On the other hand in ‘*my boss raised my salary*  $\rightarrow$  *my boss constructed my salary*’ the context {‘*my boss*’, ‘*my salary*’} for applying ‘*raise*  $\rightarrow$  *construct*’ is invalid. Following, we split the test-set for the base Lin measure into two test-sets: (a) test-set<sub>vc</sub>, which includes all correct rule applications and incorrect ones only under valid contexts, and (b) test-set<sub>ivc</sub>, which includes again all correct rule applications but incorrect ones only under invalid contexts.

Table 5 presents the performance of each compared method on the two test sets. On test-set<sub>ivc</sub>, where context mismatches are abundant, our model outperformed all other baselines (statistically significant at  $p < 0.01$ ). In addition, this time DC slightly outperformed CI. This result more explicitly shows the advantages of integrating word-level and context-sensitive topic-level similarities for differentiating valid and invalid contexts for rule applications. Yet, many invalid rule applications occur under valid contexts due to inherently incorrect rules, and we want to make sure that also in this scenario our model does not fall behind the context-insensitive measure. Indeed, on test-set<sub>vc</sub>, in which context mismatches are rare, our algorithm is still better than the original measure, indicating that WT can be safely applied to distributional similarity measures without concerns of reduced performance in different context scenarios.

	test-set <sub>ivc</sub>	test-set <sub>vc</sub>
Size (valid:invalid)	432 (266:166)	645 (266:379)
CI	0.780	0.587
DC	0.796	0.498
SC	0.779	0.512
WT	<b>0.854</b>	<b>0.621</b>

Table 5: MAP results for the two split Lin test-sets.

## 6 Discussion and Future Work

This paper addressed the problem of computing context-sensitive reliability scores for predicate inference rules. In particular, we proposed a novel scheme that applies over any base distributional similarity measure which operates at the word level, and computes a single context-insensitive score for a rule. Based on such a measure, our scheme constructs a context-sensitive similarity measure that computes a reliability score for predicate inference rules applications in the context of given arguments.

The contextualization of the base similarity score was obtained using a topic-level LDA model, which was used in a novel way. First, it provides a topic bias for learning separate per-topic word-level similarity scores between predicates. Then, given a specific candidate rule application, the LDA model is used to infer the topic distribution relevant to the context specified by the given arguments. Finally, the context-sensitive rule application score is computed as a weighted average of the per-topic word-level similarity scores, which are weighed according to the inferred topic distribution.

While most works on context-insensitive predicate inference rules, such as DIRT (Lin and Pantel, 2001), are based on word-level similarity measures, almost all prior models addressing context-sensitive predicate inference rules are based on topic models (except for (Pantel et al., 2007), which was outperformed by later models). We therefore focused on comparing the performance of our two-level scheme with state-of-the-art prior topic-level and word-level models of distributional similarity, over a random sample of inference rule applications. Under this natural setting, the two-level scheme consistently outperformed both types of models when tested with three different base similarity measures. Notably, our model shows stable performance over a large subset of the data

where context sensitivity is rare, while topic-level models tend to underperform in such cases compared to the base context-insensitive methods.

Our work is closely related to another research line that addresses lexical similarity and substitution scenarios in context. While we focus on lexical-syntactic predicate templates and instantiations of their argument slots as context, lexical similarity methods consider various lexical units that are not necessarily predicates, with their context typically being the collection of words in a window around them.

Various approaches have been proposed to address lexical similarity. A number of works are based on a compositional semantics approach, where a prior representation of a target lexical unit is composed with the representations of words in its given context (Mitchell and Lapata, 2008; Erk and Padó, 2008; Thater et al., 2010). Other works (Erk and Padó, 2010; Reisinger and Mooney, 2010) use a rather large word window around target words and compute similarities between clusters comprising instances of word windows. In addition, (Dinu and Lapata, 2010a) adapted the predicate inference topic model from (Dinu and Lapata, 2010b) to compute lexical similarity in context.

A natural extension of our work would be to extend our two level model to accommodate context-sensitive lexical similarity. For this purpose we will need to redefine the scope of context in our model, and adapt our method to compute context-biased lexical similarities accordingly. Then we will also be able to evaluate our model on the Lexical Substitution Task (McCarthy and Navigli, 2007), which has been commonly used in recent years as a benchmark for context-sensitive lexical similarity models.

In a different NLP task, Eidelman et al. (2012) utilize a similar approach to ours for improving the performance of statistical machine translation (SMT). They learn an LDA model on the source language side of the training corpus with the purpose of identifying implicit sub-domains. Then they utilize the distribution over topics inferred for each document in their corpus to compute separate per-topic translation probability tables. Finally, they train a classifier to translate a given target word based on these tables and the inferred topic distribution of the given document in which the target word appears. A notable difference be-

tween our approach and theirs is that we use predicate pseudo-documents consisting of argument instantiations to learn our LDA model, while Eidelman et al. use the real documents in a corpus. We believe that combining these two approaches may improve performance for both textual inference and SMT and plan to experiment with this direction in future work.

## Acknowledgments

This work was partially supported by the Israeli Ministry of Science and Technology grant 3-8705, the Israel Science Foundation grant 880/12, and the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT).

## References

- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *ACL*.
- Rahul Bhagat, Patrick Pantel, Eduard Hovy, and Marina Rey. 2007. Ledir: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of EMNLP-CoNLL*.
- David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Lecture Notes in Computer Science*, volume 3944, pages 177–190.
- Georgiana Dinu and Mirella Lapata. 2010a. Measuring distributional similarity in context. In *Proceedings of EMNLP*.
- Georgiana Dinu and Mirella Lapata. 2010b. Topic models for meaning similarity in context. In *Proceedings of COLING: Posters*.
- Georgiana Dinu and Rui Wang. 2009. Inference rules and their application to recognizing textual entailment. In *Proceedings EACL*.
- Vladimir Eidelman, Jordan Boyd-Graber, and Philip Resnik. 2012. Topic models for dynamic translation model adaptation. In *Proceedings of the ACL conference short papers*.
- Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of EMNLP*.
- Katrin Erk and Sebastian Padó. 2010. Exemplar-based models for word meaning in context. In *Proceedings of the ACL conference short papers*.

- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of EMNLP*.
- Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(4):359–389.
- Dekang Lin and Patrick Pantel. 2001. DIRT – discovery of inference rules from text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2001*.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL*.
- Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.
- Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of SemEval*.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*.
- Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Patty: A taxonomy of relational patterns with semantic types. *EMNLP12*.
- Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. ISP: Learning inferential selectional preferences. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*.
- Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL*.
- Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.
- Alan Ritter, Oren Etzioni, et al. 2010. A latent dirichlet allocation method for selectional preferences. In *Proceedings of ACL*.
- Stefan Schoenmackers, Jesse Davis, Oren Etzioni, and Daniel Weld. 2010. Learning first-order horn clauses from web text. In *Proceedings of EMNLP*.
- Diarmuid O Séaghdha. 2010. Latent variable models of selectional preference. In *Proceedings of ACL*.
- Satoshi Sekine. 2005. Automatic paraphrase discovery based on context and keywords between ne pairs. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.
- Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*.
- Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of COLING*.
- Idan Szpektor, Ido Dagan, Roy Bar-Haim, and Jacob Goldberger. 2008. Contextual preferences. In *Proceedings of ACL-08: HLT*.
- Stefan Thater, Hagen Fürstenaun, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of ACL*.
- Naomi Zeichner, Jonathan Berant, and Ido Dagan. 2012. Crowdsourcing inference-rule evaluation. In *Proceedings of ACL (short papers)*.

# Align, Disambiguate and Walk: A Unified Approach for Measuring Semantic Similarity

Mohammad Taher Pilehvar, David Jurgens and Roberto Navigli

Department of Computer Science

Sapienza University of Rome

{pilehvar, jurgens, navigli}@di.uniroma1.it

## Abstract

Semantic similarity is an essential component of many Natural Language Processing applications. However, prior methods for computing semantic similarity often operate at different levels, e.g., single words or entire documents, which requires adapting the method for each data type. We present a unified approach to semantic similarity that operates at multiple levels, all the way from comparing word senses to comparing text documents. Our method leverages a common probabilistic representation over word senses in order to compare different types of linguistic data. This unified representation shows state-of-the-art performance on three tasks: semantic textual similarity, word similarity, and word sense coarsening.

## 1 Introduction

Semantic similarity is a core technique for many topics in Natural Language Processing such as Textual Entailment (Berant et al., 2012), Semantic Role Labeling (Fürstenu and Lapata, 2012), and Question Answering (Surdeanu et al., 2011). For example, textual similarity enables relevant documents to be identified for information retrieval (Hliaoutakis et al., 2006), while identifying similar words enables tasks such as paraphrasing (Glickman and Dagan, 2003), lexical substitution (McCarthy and Navigli, 2009), lexical simplification (Biran et al., 2011), and Web search result clustering (Di Marco and Navigli, 2013).

Approaches to semantic similarity have often operated at separate levels: methods for word similarity are rarely applied to documents or even single sentences (Budanitsky and Hirst, 2006; Radinsky et al., 2011; Halawi et al., 2012), while document-based similarity methods require more

linguistic features, which often makes them inapplicable at the word or microtext level (Salton et al., 1975; Maguitman et al., 2005; Elsayed et al., 2008; Turney and Pantel, 2010). Despite the potential advantages, few approaches to semantic similarity operate at the sense level due to the challenge in sense-tagging text (Navigli, 2009); for example, none of the top four systems in the recent SemEval-2012 task on textual similarity compared semantic representations that incorporated sense information (Agirre et al., 2012).

We propose a unified approach to semantic similarity across multiple representation levels from senses to documents, which offers two significant advantages. First, the method is applicable independently of the input type, which enables meaningful similarity comparisons across different scales of text or lexical levels. Second, by operating at the sense level, a unified approach is able to identify the semantic similarities that exist independently of the text's lexical forms and any semantic ambiguity therein. For example, consider the sentences:

- t1. *A manager fired the worker.*
- t2. *An employee was terminated from work by his boss.*

A surface-based approach would label the sentences as dissimilar due to the minimal lexical overlap. However, a sense-based representation enables detection of the similarity between the meanings of the words, e.g., *fire* and *terminate*. Indeed, an accurate, sense-based representation is essential for cases where different words are used to convey the same meaning.

The contributions of this paper are threefold. First, we propose a new unified representation of the meaning of an arbitrarily-sized piece of text, referred to as a **lexical item**, using a sense-based probability distribution. Second, we propose a novel alignment-based method for word sense dis-

ambiguation during semantic comparison. Third, we demonstrate that this single representation can achieve state-of-the-art performance on three similarity tasks, each operating at a different lexical level: (1) surpassing the highest scores on the SemEval-2012 task on textual similarity (Agirre et al., 2012) that compares sentences, (2) achieving a near-perfect performance on the TOEFL synonym selection task proposed by Landauer and Dumais (1997), which measures word pair similarity, and also obtaining state-of-the-art performance in terms of the correlation with human judgments on the RG-65 dataset (Rubenstein and Goodenough, 1965), and finally (3) surpassing the performance of Snow et al. (2007) in a sense-coarsening task that measures sense similarity.

## 2 A Unified Semantic Representation

We propose a representation of any lexical item as a distribution over a set of word senses, referred to as the item’s **semantic signature**. We begin with a formal description of the representation at the sense level (Section 2.1). Following this, we describe our alignment-based disambiguation algorithm which enables us to produce sense-based semantic signatures for those lexical items (e.g., words or sentences) which are not sense annotated (Section 2.2). Finally, we propose three methods for comparing these signatures (Section 2.3). As our sense inventory, we use WordNet 3.0 (Fellbaum, 1998).

### 2.1 Semantic Signatures

The WordNet ontology provides a rich network structure of semantic relatedness, connecting senses directly with their hypernyms, and providing information on semantically similar senses by virtue of their nearby locality in the network. Given a particular node (sense) in the network, repeated random walks beginning at that node will produce a frequency distribution over the nodes in the graph visited during the walk. To extend beyond a single sense, the random walk may be initialized and restarted from a set of senses (seed nodes), rather than just one; this multi-seed walk produces a multinomial distribution over all the senses in WordNet with higher probability assigned to senses that are frequently visited from the seeds. Prior work has demonstrated that multinomials generated from random walks over WordNet can be successfully applied to linguistic tasks such as word similarity (Hughes and Ramage,

2007; Agirre et al., 2009), paraphrase recognition, textual entailment (Ramage et al., 2009), and pseudoword generation (Pilehvar and Navigli, 2013).

Formally, we define the semantic signature of a lexical item as the multinomial distribution generated from the random walks over WordNet 3.0 where the set of seed nodes is the set of senses present in the item. This representation encompasses both when the item is itself a single sense and when the item is a sense-tagged sentence.

To construct each semantic signature, we use the iterative method for calculating topic-sensitive PageRank (Haveliwala, 2002). Let  $M$  be the adjacency matrix for the WordNet network, where edges connect senses according to the relations defined in WordNet (e.g., hypernymy and meronymy). We further enrich  $M$  by connecting a sense with all the other senses that appear in its disambiguated gloss.<sup>1</sup> Let  $\vec{v}^{(0)}$  denote the probability distribution for the starting location of the random walker in the network. Given the set of senses  $S$  in a lexical item, the probability mass of  $\vec{v}^{(0)}$  is uniformly distributed across the senses  $s_i \in S$ , with the mass for all  $s_j \notin S$  set to zero. The PageRank may then be computed using:

$$\vec{v}^{(t)} = (1 - \alpha) M \vec{v}^{(t-1)} + \alpha \vec{v}^{(0)} \quad (1)$$

where at each iteration the random walker may jump to any node  $s_i \in S$  with probability  $\alpha/|S|$ . We follow standard convention and set  $\alpha$  to 0.15. We repeat the operation in Eq. 1 for 30 iterations, which is sufficient for the distribution to converge. The resulting probability vector  $\vec{v}^{(t)}$  is the semantic signature of the lexical item, as it has aggregated its senses’ similarities over the entire graph. For our semantic signatures we used the UKB<sup>2</sup> off-the-shelf implementation of topic-sensitive PageRank.

### 2.2 Alignment-Based Disambiguation

Commonly, semantic comparisons are between word pairs or sentence pairs that do not have their lexical content sense-annotated, despite the potential utility of sense annotation in making semantic comparisons. However, traditional forms of word sense disambiguation are difficult for short texts and single words because little or no contextual information is present to perform the disambiguation task. Therefore, we propose a novel

<sup>1</sup><http://wordnet.princeton.edu>

<sup>2</sup><http://ixa2.si.ehu.es/ukb/>

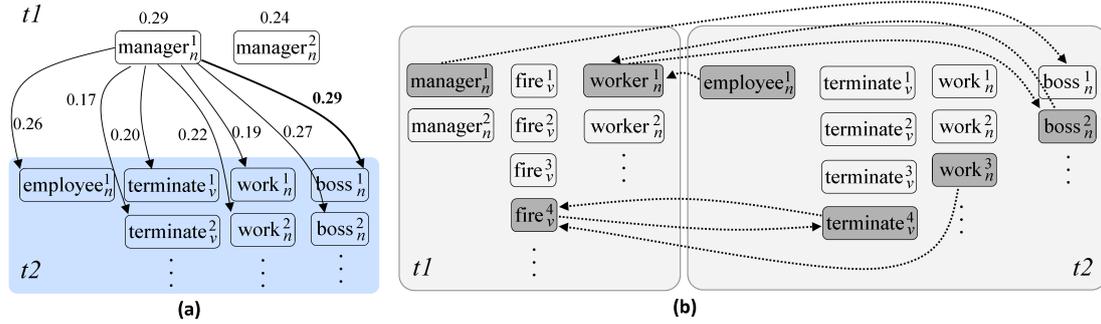


Figure 1: (a) Example alignments of the first sense of term *manager* (in sentence  $t1$ ) to the two first senses of the word types in sentence  $t2$ , along with the similarity of the two senses’ semantic signatures; (b) Alignments which maximize the similarities across words in  $t1$  and  $t2$  (the source side of an alignment is taken as the disambiguated sense of its corresponding word).

alignment-based sense disambiguation that leverages the content of the paired item in order to disambiguate each element. Leveraging the paired item enables our approach to disambiguate where traditional sense disambiguation methods can not due to insufficient context.

We view sense disambiguation as an **alignment problem**. Given two arbitrarily ordered texts, we seek the semantic alignment that maximizes the similarity of the senses of the context words in both texts. To find this maximum we use an alignment procedure which, for each word type  $w_i$  in item  $T_1$ , assigns  $w_i$  to the sense that has the maximal similarity to any sense of the word types in the compared text  $T_2$ . Algorithm 1 formalizes the alignment process, which produces a sense disambiguated representation as a result. Senses are compared in terms of their semantic signatures, denoted as function  $\mathcal{R}$ . We consider multiple definitions of  $\mathcal{R}$ , defined later in Section 2.3.

As a part of the disambiguation procedure, we leverage the one sense per discourse heuristic of Yarowsky (1995); given all the word types in two compared lexical items, each type is assigned a single sense, even if it is used multiple times. Additionally, if the same word type appears in both sentences, both will always be mapped to the same sense. Although such a sense assignment is potentially incorrect, assigning both types to the same sense results in a representation that does no worse than a surface-level comparison.

We illustrate the alignment-based disambiguation procedure using the two example sentences  $t1$  and  $t2$  given in Section 1. Figure 1(a) illustrates example alignments of the first sense of *manager* to the first two senses of the word types in sentence  $t2$  along with the similarity of the two senses’

---

### Algorithm 1 Alignment-based Sense Disambiguation

---

**Input:**  $T_1$  and  $T_2$ , the sets of word types being compared

**Output:**  $P$ , the set of disambiguated senses for  $T_1$

- 1:  $P \leftarrow \emptyset$
  - 2: **for each** token  $t_i \in T_1$
  - 3:    $max\_sim \leftarrow 0$
  - 4:    $best\_s_i \leftarrow null$
  - 5:   **for each** token  $t_j \in T_2$
  - 6:     **for each**  $s_i \in Senses(t_i), s_j \in Senses(t_j)$
  - 7:        $sim \leftarrow \mathcal{R}(s_i, s_j)$
  - 8:       **if**  $sim > max\_sim$  **then**
  - 9:          $max\_sim = sim$
  - 10:         $best\_s_i = s_i$
  - 11:    $P \leftarrow P \cup \{best\_s_i\}$
  - 12: **return**  $P$
- 

semantic signatures. For the senses of *manager*, sense  $manager_n^1$  obtains the maximal similarity value to  $boss_n^1$  among all the possible pairings of the senses for the word types in sentence  $t2$ , and as a result is selected as the sense labeling for *manager* in sentence  $t1$ .<sup>3</sup> Figure 1(b) shows the final, maximally-similar sense alignment of the word types in  $t1$  and  $t2$ . The resulting alignment produces the following sets of senses:

$$P_{t1} = \{manager_n^1, fire_v^4, worker_n^1\}$$

$$P_{t2} = \{employee_n^1, terminate_v^4, work_n^3, boss_n^2\}$$

where  $P_x$  denotes the corresponding set of senses of sentence  $x$ .

### 2.3 Semantic Signature Similarity

**Cosine Similarity.** In order to compare semantic signatures, we adopt the *Cosine* similarity measure as a baseline method. The measure is computed by treating each multinomial as a vector and then calculating the normalized dot product of the two signatures’ vectors.

<sup>3</sup>We follow Navigli (2009) and denote with  $w_p^i$  the  $i$ -th sense of  $w$  in WordNet with part of speech  $p$ .

However, a semantic signature is, in essence, a weighted ranking of the importance of WordNet senses for each lexical item. Given that the WordNet graph has a non-uniform structure, and also given that different lexical items may be of different sizes, the magnitudes of the probabilities obtained may differ significantly between the two multinomial distributions. Therefore, for computing the similarity of two signatures, we also consider two nonparametric methods that use the ranking of the senses, rather than their probability values, in the multinomial.

**Weighted Overlap.** Our first measure provides a nonparametric similarity by comparing the similarity of the rankings for intersection of the senses in both semantic signatures. However, we additionally weight the similarity such that differences in the highest ranks are penalized more than differences in lower ranks. We refer to this measure as the *Weighted Overlap*. Let  $S$  denote the intersection of all senses with non-zero probability in both signatures and  $r_i^j$  denote the rank of sense  $s_i \in S$  in signature  $j$ , where rank 1 denotes the highest rank. The sum of the two ranks  $r_i^1$  and  $r_i^2$  for a sense is then inverted, which (1) weights higher ranks more and (2) when summed, provides the maximal value when a sense has the same rank in both signatures. The unnormalized weighted overlap is then calculated as  $\sum_{i=1}^{|S|} (r_i^1 + r_i^2)^{-1}$ . Then, to bound the similarity value in  $[0, 1]$ , we normalize the sum by its maximum value,  $\sum_{i=1}^{|S|} (2i)^{-1}$ , which occurs when each sense has the same rank in both signatures.

**Top- $k$  Jaccard.** Our second measure uses the ranking to identify the top- $k$  senses in a signature, which are treated as the best representatives of the conceptual associates. We hypothesize that a specific rank ordering may be attributed to small differences in the multinomial probabilities, which can lower rank-based similarities when one of the compared orderings is perturbed due to slightly different probability values. Therefore, we consider the top- $k$  senses as an unordered set, with equal importance in the signature. To compare two signatures, we compute the Jaccard Index of the two signatures’ sets:

$$\mathcal{R}_{Jac}(U_k, V_k) = \frac{|U_k \cap V_k|}{|U_k \cup V_k|} \quad (2)$$

where  $U_k$  denotes the set of  $k$  senses with the highest probability in the semantic signature  $U$ .

Dataset	MSRvid	MSRpar	SMTeuroparl	OnWN	SMTnews
Training	750	750	734	-	-
Test	750	750	459	750	399

Table 1: Statistics of the provided datasets for the SemEval-2012 Semantic Textual Similarity task.

### 3 Experiment 1: Textual Similarity

Measuring semantic similarity of textual items has applications in a wide variety of NLP tasks. As our benchmark, we selected the recent SemEval-2012 task on Semantic Textual Similarity (STS), which was concerned with measuring the semantic similarity of sentence pairs. The task received considerable interest by facilitating a meaningful comparison between approaches.

#### 3.1 Experimental Setup

**Data.** We follow the experimental setup used in the STS task (Agirre et al., 2012), which provided five test sets, two of which had accompanying training data sets for tuning system performance. Each sentence pair in the datasets was given a score from 0 to 5 (low to high similarity) by human judges, with a high inter-annotator agreement of around 0.90 when measured using the Pearson correlation coefficient. Table 1 lists the number of sentence pairs in training and test portions of each dataset.

**Comparison Systems.** The top-ranking participating systems in the SemEval-2012 task were generally supervised systems utilizing a variety of lexical resources and similarity measurement techniques. We compare our results against the top three systems of the 88 submissions: TLsim and TLsyn, the two systems of Šarić et al. (2012), and the UKP2 system (Bär et al., 2012). UKP2 utilizes extensive resources among which are a Distributional Thesaurus computed on 10M dependency-parsed English sentences. In addition, the system utilizes techniques such as Explicit Semantic Analysis (Gabrilovich and Markovitch, 2007) and makes use of resources such as Wiktionary and Wikipedia, a lexical substitution system based on supervised word sense disambiguation (Biemann, 2013), and a statistical machine translation system. The TLsim system uses the New York Times Annotated Corpus, Wikipedia, and Google Book Ngrams. The TLsyn system also uses Google Book Ngrams, as well as dependency parsing and named entity recognition.

Ranking			System	Overall			Dataset-specific				
ALL	ALLnrm	Mean		ALL	ALLnrm	Mean	Mpar	Mvid	SMTe	OnWN	SMTn
1	1	1	ADW	<b>0.866</b>	<b>0.871</b>	<b>0.711</b>	0.694	<b>0.887</b>	<b>0.555</b>	<b>0.706</b>	<b>0.604</b>
2	3	2	UKP2	0.824	0.858	0.677	0.683	0.873	0.528	0.664	0.493
3	4	6	TLsyn	0.814	0.857	0.660	0.698	0.862	0.361	0.704	0.468
4	2	3	TLsim	0.813	0.864	0.675	<b>0.734</b>	0.880	0.477	0.679	0.398

Table 2: Performance of our system (ADW) and the 3 top-ranking participating systems (out of 88) in the SemEval-2012 Semantic Textual Similarity task. Rightmost columns report the corresponding Pearson correlation  $r$  for individual datasets, i.e., MSRpar (Mpar), MSRvid (Mvid), SMTEuoparl (SMTe), OnWN (OnWN) and SMTnews (SMTn). We also provide scores according to the three official evaluation metrics (i.e., ALL, ALLnrm, and Mean). Rankings are also presented based on the three metrics.

**System Configuration.** Here we describe the configuration of our approach, which we have called Align, Disambiguate and Walk (ADW). The STS task uses human similarity judgments on an ordinal scale from 0 to 5. Therefore, in ADW we adopted a similar approach to generating similarity values to that adopted by other participating systems, whereby a supervised system is trained to combine multiple similarity judgments to produce a final rating consistent with the human annotators. We utilized the WEKA toolkit (Hall et al., 2009) to train a Gaussian Processes regression model for each of the three training sets (cf. Table 1). The features discussed hereafter were considered in our regression model.

**Main features.** We used the scores calculated using all three of our semantic signature comparison methods as individual features. Although the *Jaccard* comparison is parameterized, we avoided tuning and instead used four features for distinct values of  $k$ : 250, 500, 1000, and 2500.

**String-based features.** Additionally, because the texts often contain named entities which are not present in WordNet, we incorporated the similarity values produced by four string-based measures, which were used by other teams in the STS task: (1) *longest common substring* which takes into account the length of the longest overlapping contiguous sequence of characters (substring) across two strings (Gusfield, 1997), (2) *longest common subsequence* which, instead, finds the longest overlapping subsequence of two strings (Allison and Dix, 1986), (3) *Greedy String Tiling* which allows reordering in strings (Wise, 1993), and (4) the character/word  $n$ -gram similarity proposed by Barrón-Cedeño et al. (2010).

We followed Šarić et al. (2012) and used the models trained on the SMTEuoparl and MSRpar datasets for testing on the SMTnews and OnWN test sets, respectively.

### 3.2 STS Results

Three evaluation metrics are provided by the organizers of the SemEval-2012 STS task, all of which are based on Pearson correlation  $r$  of human judgments with system outputs: (1) the correlation value for the concatenation of all five datasets (ALL), (2) a correlation value obtained on a concatenation of the outputs, separately normalized by least square (ALLnrm), and (3) the weighted average of Pearson correlations across datasets (Mean). Table 2 shows the scores obtained by ADW for the three evaluation metrics, as well as the Pearson correlation values obtained on each of the five test sets (rightmost columns). We also show the results obtained by the three top-ranking participating systems (i.e., UKP2, TLsim, and TLsyn). The leftmost three columns show the system rankings according to the three metrics.

As can be seen from Table 2, our system (ADW) outperforms all the 88 participating systems according to all the evaluation metrics. Our system shows a statistically significant improvement on the SMTnews dataset, with an increase in the Pearson correlation of over 0.10. MSRpar (MPar) is the only dataset in which TLsim (Šarić et al., 2012) achieves a higher correlation with human judgments. Named entity features used by the TLsim system could be the reason for its better performance on the MSRpar dataset, which contains a large number of named entities.

### 3.3 Similarity Measure Analysis

To gain more insight into the impact of our alignment-based disambiguation approach, we carried out a 10-fold cross-validation on the three training datasets (cf. Table 1) using the systems described hereafter.

**ADW-MF.** To build this system, we utilized our main features only; i.e., we did not make use of additional string-based features.

**DW.** Similarly to ADW-MF, this system utilized the main features only. In DW, however, we replaced our alignment-based disambiguation phase with a random walk-based WSD system that disambiguated the sentences separately, without performing any alignment. As our WSD system, we used UKB, a state-of-the-art knowledge-based WSD system that is based on the same topic-sensitive PageRank algorithm used by our approach. UKB initializes the algorithm from all senses of the words in the context of a word to be disambiguated. It then picks the most relevant sense of the word according to the resulting probability vector. As the lexical knowledge base of UKB, we used the same semantic network as that utilized by our approach for calculating semantic signatures.

Table 3 lists the performance values of the two above-mentioned systems on the three training sets in terms of Pearson correlation. In addition, we present in the table correlation scores for four other similarity measures reported by Bär et al. (2012):

- Pairwise Word Similarity that comprises of a set of WordNet-based similarity measures proposed by Resnik (1995), Jiang and Conrath (1997), and Lin (1998b). The aggregation strategy proposed by Corley and Mihalcea (2005) has been utilized for extending these word-to-word similarity measures for calculating text-to-text similarities.
- Explicit Semantic Analysis (Gabrilovich and Markovitch, 2007) where the high-dimensional vectors are obtained on WordNet, Wikipedia and Wiktionary.
- Distributional Thesaurus where a similarity score is computed similarly to that of Lin (1998a) using a distributional thesaurus obtained from a 10M dependency-parsed sentences of English newswire.
- Character  $n$ -grams which were also used as one of our additional features.

As can be seen from Table 3, our alignment-based disambiguation approach (ADW-MF) is better suited to the task than a conventional WSD approach (DW). Another interesting point is the high scores achieved by the Character  $n$ -grams

Similarity measure	Dataset		
	Mpar	Mvid	SMTe
DW	0.448	0.820	0.660
ADW-MF	0.485	<b>0.842</b>	<b>0.721</b>
Explicit Semantic Analysis	0.427	0.781	0.619
Pairwise Word Similarity	0.564	0.835	0.527
Distributional Thesaurus	0.494	0.481	0.365
Character $n$ -grams	<b>0.658</b>	0.771	0.554

Table 3: Performance of our main-feature system with conventional WSD (DW) and with the alignment-based disambiguation approach (ADW-MF) vs. four other similarity measures, using 10-fold cross validation on the training datasets MSRpar (Mpar), MSRvid (Mvid), and SMTeuroparl (SMTe).

measure. This confirms that string-based methods are strong baselines for semantic textual similarity. Except for the MSRpar (Mpar) dataset, our system (ADW-MF) outperforms all other similarity measures. The scores obtained by Explicit Semantic Analysis and Distributional Thesaurus are not competitive on any dataset. On the other hand, Pairwise Word Similarity achieves a high performance on MSRpar and MSRvid datasets, but performs surprisingly low on the SMTeuroparl dataset.

## 4 Experiment 2: Word Similarity

We now proceed from the sentence level to the word level. Word similarity has been a key problem for lexical semantics, with significant efforts being made by approaches in distributional semantics to accurately identify synonymous words (Turney and Pantel, 2010). Different evaluation methods exist in the literature for evaluating the performance of a word-level semantic similarity measure; we adopted two well-established benchmarks: synonym recognition and correlating word similarity judgments with those from human annotators.

For synonym recognition, we used the TOEFL dataset created by Landauer and Dumais (1997). The dataset consists of 80 multiple-choice synonym questions from the TOEFL test; a word is paired with four options, one of which is a valid synonym. Test takers with English as a second language averaged 64.5% correct. Despite multiple approaches, only recently has the test been answered perfectly (Bullinaria and Levy, 2012), underscoring the challenge of synonym recognition.

Approach	Accuracy
PPMIC (Bullinaria and Levy, 2007)	85.00%
GLSA (Matveeva et al., 2005)	86.25%
LSA (Rapp, 2003)	92.50%
ADW <sub>Jac</sub>	93.75±2.5%
ADW <sub>WO</sub>	95.00%
ADW <sub>Cos</sub>	96.25%
PR (Turney et al., 2003)	97.50%
PCCP (Bullinaria and Levy, 2012)	100.00%

Table 4: Accuracy on the 80-question TOEFL Synonym test. ADW<sub>Jac</sub>, ADW<sub>WO</sub>, and ADW<sub>Cos</sub> correspond to results with the *Jaccard*, *Weighted Overlap* and *Cosine* signature comparison measures, respectively.

For the similarity judgment evaluation, we used as benchmark the RG-65 dataset created by Rubenstein and Goodenough (1965). The dataset contains 65 word pairs judged by 51 human subjects on a scale of 0 to 4 according to their semantic similarity. Ideally, a measure’s similarity judgments are expected to be highly correlated with those of humans. To be consistent with the previous literature (Hughes and Ramage, 2007; Agirre et al., 2009), we used Spearman’s rank correlation in our experiment.

#### 4.1 Experimental Setup

Our alignment-based sense disambiguation transforms the task of comparing individual words into that of calculating the similarity of the best-matching sense pair across the two words. As there is no training data we do not optimize the  $k$  value for computing signature similarity with the *Jaccard* index; instead, we report, for the synonym recognition and the similarity judgment evaluations, the respective range of accuracies and the average correlation obtained upon using five values of  $k$  randomly selected in the range [50, 2500]: 678, 1412, 1692, 2358, 2387.

#### 4.2 Word Similarity Results: TOEFL dataset

Table 4 lists the accuracy performance of the system in comparison to the existing state of the art on the TOEFL test. ADW<sub>WO</sub>, ADW<sub>Cos</sub>, and ADW<sub>Jac</sub> correspond to our approach when *Weighted Overlap*, *Cosine*, and *Jaccard* signature comparison measures are used, respectively. Despite not being tuned for the task, our model achieves near-perfect performance, answering all but three questions correctly with the *Cosine* measure. Among the top-performing approaches, only

Word	Synonym choices (correct in bold)			
fanciful	familiar	apparent*	<b>imaginative</b> †	logical
verbal	<b>oral</b> †	overt	fitting	verbose*
resolved	<b>settled</b> *	forgotten†	publicized	examined
percentage	volume	sample	<b>proportion</b>	profit†*
figure	list	<b>solve</b> *	divide†	express
highlight	alter†	imitate	<b>accentuate</b> *	restore

Table 5: Questions answered incorrectly by our approach. Symbols † and \* correspond to the choices of our approach with the *Weighted Overlap* and *Cosine* signature comparisons respectively. We do not include the mistakes made when the *Jaccard* measure was used as they vary with the  $k$  value.

that of Rapp (2003) uses word senses, an approach that is outperformed by our method.

The errors produced by our system were largely the result of sense locality in the WordNet network. Table 5 highlights the incorrect responses. The synonym mistakes reveal cases where senses of the two words are close in WordNet, indicating some relatedness. For example, *percentage* may be interpreted colloquially as monetary value (e.g., “give me my percentage”) and elicits the synonym of *profit* in the economic domain, which ADW incorrectly selects as a synonym.

#### 4.3 Word Similarity Results: RG-65 dataset

Table 6 shows the Spearman’s  $\rho$  rank correlation coefficients with human judgments on the RG-65 dataset. As can be seen from the Table, our approach with the *Weighted Overlap* signature comparison improves over the similar approach of Hughes and Ramage (2007) which, however, does not involve the disambiguation step and considers a word as a whole unit as represented by the set of its senses.

### 5 Experiment 3: Sense Similarity

WordNet is known to be a fine-grained sense inventory with many related word senses (Palmer et al., 2007). Accordingly, multiple approaches have attempted to identify highly similar senses in order to produce a coarse-grained sense inventory. We adopt this task as a way of evaluating our similarity measure at the sense level.

#### 5.1 Coarse-graining Background

Earlier work on reducing the polysemy of sense inventories has considered WordNet-based sense relatedness measures (Mihalcea and Moldovan, 2001) and corpus-based vector representations of

Approach	Correlation
ADW <sub>Cos</sub>	0.825
Agirre et al. (2009)	0.830
Hughes and Ramage (2007)	0.838
Zesch et al. (2008)	0.840
ADW <sub>Jac</sub>	0.841
ADW <sub>WO</sub>	0.868

Table 6: Spearman’s  $\rho$  correlation coefficients with human judgments on the RG-65 dataset. ADW<sub>Jac</sub>, ADW<sub>WO</sub>, and ADW<sub>Cos</sub> correspond to results with the *Jaccard*, *Weighted Overlap* and *Cosine* signature comparison measures respectively.

word senses (Agirre and Lopez, 2003; McCarthy, 2006). Navigli (2006) proposed an automatic approach for mapping WordNet senses to the coarse-grained sense distinctions of the Oxford Dictionary of English (ODE). The approach leverages semantic similarities in gloss definitions and the hierarchical relations between senses in the ODE to cluster WordNet senses. As current state of the art, Snow et al. (2007) developed a supervised SVM classifier that utilized, as its features, several earlier sense relatedness techniques such as those implemented in the WordNet::Similarity package (Pedersen et al., 2004). The classifier also made use of resources such as topic signatures data (Agirre and de Lacalle, 2004), the WordNet domain dataset (Magnini and Cavaglia, 2000), and the mappings of WordNet senses to ODE senses produced by Navigli (2006).

## 5.2 Experimental Setup

We benchmark the accuracy of our similarity measure in grouping word senses against those of Navigli (2006) and Snow et al. (2007) on two datasets of manually-labeled sense groupings of WordNet senses: (1) sense groupings provided as a part of the Senseval-2 English Lexical Sample WSD task (Kilgarriff, 2001) which includes nouns, verbs and adjectives; (2) sense groupings included in the OntoNotes project<sup>4</sup> (Hovy et al., 2006) for nouns and verbs. Following the evaluation methodology of Snow et al. (2007), we combine the Senseval-2 and OntoNotes datasets into a third dataset.

Snow et al. (2007) considered sense grouping as a binary classification task whereby for each word every possible pairing of senses has to be classified

<sup>4</sup>Sense groupings belong to a pre-version 1.0: <http://cemantix.org/download/sense/ontonotes-sense-groups.tar.gz>

	Onto		SE-2			Onto + SE-2	
Method	Noun	Verb	Noun	Verb	Adj	Noun	Verb
$\mathcal{R}_{Cos}$	0.406	0.522	0.450	0.465	0.484	0.441	0.485
$\mathcal{R}_{WO}$	<b>0.421</b>	<b>0.544</b>	<b>0.483</b>	<b>0.482</b>	<b>0.531</b>	<b>0.470</b>	<b>0.503</b>
$\mathcal{R}_{Jac}$	0.418	0.531	0.478	0.473	0.501	0.465	0.493
SVM	0.370	0.455	NA	NA	0.473	0.423	0.432
ODE	0.218	0.396	NA	NA	0.371	0.331	0.288

Table 7: F-score sense merging evaluation on three hand-labeled datasets: OntoNotes (Onto), Senseval-2 (SE-2), and combined (Onto+SE-2). Results are reported for all three of our signature comparison measures and also for two previous works (last two rows).

as either *merged* or *not-merged*. We constructed a simple threshold-based classifier to perform the same binary classification. To this end, we calculated the semantic similarity of each sense pair and then used a threshold value  $t$  to classify the pair as merged if similarity  $\geq t$  and not-merged otherwise. We sampled out 10% of the dataset for tuning the value of  $t$ , thus adapting our classifier to the fine granularity of the dataset. We used the same held-out instances to perform a tuning of the  $k$  value used for *Jaccard* index, over the same values of  $k$  as in Experiment 1 (cf. Section 3).

## 5.3 Sense Merging Results

For a binary classification task, we can directly calculate precision, recall and F-score by constructing a contingency table. We show in Table 7 the F-score performance of our classifier as obtained by an averaged 10-fold cross-validation. Results are presented for all three of the measures of semantic signature comparison and for the three datasets: OntoNotes, Senseval-2, and the two combined. In addition, we show in Table 7 the F-score results provided by Snow et al. (2007) for their SVM-based system and for the mapping-based approach of Navigli (2006), denoted by ODE.

Table 7 shows that our methodology yields improvements over previous work on both datasets and for all parts of speech, irrespective of the semantic signature comparison method used. Among the three methods, *Weighted Overlap* achieves the best performance, which demonstrates that our transformation of semantic signatures into ordered lists of concepts and calculating similarity by rank comparison has been helpful.

## 6 Related Work

Due to the wide applicability of semantic similarity, significant efforts have been made at different lexical levels. Early work on document-level similarity was driven by information retrieval. Vector space methods provided initial successes (Salton et al., 1975), but often suffer from data sparsity when using small documents, or when documents use different word types, as in the case of paraphrases. Later efforts such as LSI (Deerwester et al., 1990), PLSA (Hofmann, 2001) and Topic Models (Blei et al., 2003; Steyvers and Griffiths, 2007) overcame these sparsity issues using dimensionality reduction techniques or modeling the document using latent variables. However, such methods were still most suitable for comparing longer texts. Complementary approaches have been developed specifically for comparing shorter texts, such as those used in the SemEval-2012 STS task (Agirre et al., 2012). Most similar to our approach are the methods of Islam and Inkpen (2008) and Corley and Mihalcea (2005), who performed a word-to-word similarity alignment; however, they did not operate at the sense level. Ramage et al. (2009) used a similar semantic representation of short texts from random walks on WordNet, which was applied to paraphrase recognition and textual entailment. However, unlike our approach, their method does not perform sense disambiguation prior to building the representation and therefore potentially suffers from ambiguity.

A significant amount of effort has also been put into measuring similarity at the word level, frequently by approaches that use distributional semantics (Turney and Pantel, 2010). These methods use contextual features to represent semantics at the word level, whereas our approach represents word semantics at the sense level. Most similar to our approach are those of Agirre et al. (2009) and Hughes and Ramage (2007), which represent word meaning as the multinomials produced from random walks on the WordNet graph. However, unlike our approach, neither of these disambiguates the two words being compared, which potentially conflates the meanings and lowers the similarity judgment.

Measures of sense relatedness have frequently leveraged the structural properties of WordNet (e.g., path lengths) to compare senses. Budanitsky and Hirst (2006) provided a survey of such WordNet-based measures. The main drawback

with these approaches lies in the WordNet structure itself, where frequently two semantically similar senses are distant in the WordNet hierarchy. Possible solutions include relying on wider-coverage networks such as WikiNet (Nastase and Strube, 2013) or multilingual ones such as BabelNet (Navigli and Ponzetto, 2012b). Fewer works have focused on measuring the similarity – as opposed to relatedness – between senses. The topic signatures method of Agirre and Lopez (2003) represents each sense as a vector over corpus-derived features in order to build comparable sense representations. However, topic signatures often produce lower quality representations due to sparsity in the local structure of WordNet, especially for rare senses. In contrast, the random walk used in our approach provides a denser, and thus more comparable, representation for all WordNet senses.

## 7 Conclusions

This paper presents a unified approach for computing semantic similarity at multiple lexical levels, from word senses to texts. Our method leverages a common probabilistic representation at the sense level for all types of linguistic data. We demonstrate that our semantic representation achieves state-of-the-art performance in three experiments using semantic similarity at different lexical levels (i.e., sense, word, and text), surpassing the performance of previous similarity measures that are often specifically targeted for each level.

In future work, we plan to explore the impact of the sense inventory-based network used in our semantic signatures. Specifically, we plan to investigate higher coverage inventories such as BabelNet (Navigli and Ponzetto, 2012a), which will handle texts with named entities and rare senses that are not in WordNet, and will also enable cross-lingual semantic similarity. Second, we plan to evaluate our method on larger units of text and formalize comparison methods between different lexical levels.

## Acknowledgments

The authors gratefully acknowledge the support of the ERC Starting Grant MultiJEDI No. 259234.  

We would like to thank Sameer S. Pradhan for providing us with an earlier version of the OntoNotes dataset.

## References

- Eneko Agirre and Oier Lopez de Lacalle. 2004. Publicly available topic signatures for all WordNet nominal senses. In *Proceedings of LREC*, pages 1123–1126, Lisbon, Portugal.
- Eneko Agirre and Oier Lopez. 2003. Clustering WordNet word senses. In *Proceedings of RANLP*, pages 121–130, Borovets, Bulgaria.
- Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of NAACL*, pages 19–27, Boulder, Colorado.
- Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of SemEval-2012*, pages 385–393, Montreal, Canada.
- Lloyd Allison and Trevor I. Dix. 1986. A bit-string longest-common-subsequence algorithm. *Information Processing Letters*, 23(6):305–310.
- Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of SemEval-2012*, pages 435–440, Montreal, Canada.
- Alberto Barrón-Cedeño, Paolo Rosso, Eneko Agirre, and Gorka Labaka. 2010. Plagiarism detection across distant language pairs. In *Proceedings of COLING*, pages 37–45, Beijing, China.
- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2012. Learning entailment relations by global graph structure optimization. *Computational Linguistics*, 38(1):73–111.
- Chris Biemann. 2013. Creating a system for lexical substitutions from scratch using crowdsourcing. *Language Resources and Evaluation*, 47(1):97–122.
- Or Biran, Samuel Brody, and Noémie Elhadad. 2011. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of ACL*, pages 496–501, Portland, Oregon.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022.
- Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13–47.
- John A. Bullinaria and Joseph P. Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, (3):510.
- John A. Bullinaria and Joseph P. Levy. 2012. Extracting semantic representations from word co-occurrence statistics: stop-lists, stemming, and SVD. *Behavior Research Methods*, 44:890–907.
- Courtney Corley and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, Ann Arbor, Michigan.
- Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of American Society for Information Science*, 41(6):391–407.
- Antonio Di Marco and Roberto Navigli. 2013. Clustering and diversifying Web search results with graph-based Word Sense Induction. *Computational Linguistics*, 39(3).
- Tamer Elsayed, Jimmy Lin, and Douglas W. Oard. 2008. Pairwise document similarity in large collections with MapReduce. In *Proceedings of ACL-HLT*, pages 265–268, Columbus, Ohio.
- Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Database*. MIT Press, Cambridge, MA.
- Hagen Fürstenaу and Mirella Lapata. 2012. Semi-supervised Semantic Role Labeling via structural alignment. *Computational Linguistics*, 38(1):135–171.
- Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of IJCAI*, pages 1606–1611, Hyderabad, India.
- Oren Glickman and Ido Dagan. 2003. Acquiring lexical paraphrases from a single corpus. In *Proceedings of RANLP*, pages 81–90, Borovets, Bulgaria.
- Dan Gusfield. 1997. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press.
- Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of KDD*, pages 1406–1414, Beijing, China.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.
- Taher H. Haveliwala. 2002. Topic-sensitive PageRank. In *Proceedings of WWW*, pages 517–526, Hawaii, USA.
- Angelos Hliaoutakis, Giannis Varelas, Epimenidis Voutsakis, Euripides GM Petrakis, and Evangelos Milios. 2006. Information retrieval by semantic similarity. *International Journal on Semantic Web and Information Systems*, 2(3):55–73.
- Thomas Hofmann. 2001. Unsupervised Learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42(1):177–196.
- Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of NAACL*, pages 57–60, NY, USA.
- Thad Hughes and Daniel Ramage. 2007. Lexical semantic relatedness with random graph walks. In *Proceedings of EMNLP-CoNLL*, pages 581–589, Prague, Czech Republic.
- Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data*, 2(2):10:1–10:25.
- Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of ROCLING X*, pages 19–30, Taiwan.

- Adam Kilgarriff. 2001. English lexical sample task description. In *Proceedings of Senseval*, pages 17–20, Toulouse, France.
- Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review; Psychological Review*, 104(2):211.
- Dekang Lin. 1998a. Automatic retrieval and clustering of similar words. In *Proceedings of COLING*, pages 768–774, Montreal, Quebec, Canada.
- Dekang Lin. 1998b. An information-theoretic definition of similarity. In *Proceedings of ICML*, pages 296–304, San Francisco, CA.
- Bernardo Magnini and Gabriela Cavaglia. 2000. Integrating subject field codes into WordNet. In *Proceedings of LREC*, pages 1413–1418, Athens, Greece.
- Ana G. Maguitman, Filippo Menczer, Heather Roinestad, and Alessandro Vespignani. 2005. Algorithmic detection of semantic similarity. In *Proceedings of WWW*, pages 107–116, Chiba, Japan.
- Irina Matveeva, Gina-Anne Levow, Ayman Farahat, and Christiaan Royer. 2005. Terms representation with generalized latent semantic analysis. In *Proceedings of RANLP*, Borovets, Bulgaria.
- Diana McCarthy and Roberto Navigli. 2009. The English lexical substitution task. *Language Resources and Evaluation*, 43(2):139–159.
- Diana McCarthy. 2006. Relating WordNet senses for word sense disambiguation. In *Proceedings of the Workshop on Making Sense of Sense at EACL-06*, pages 17–24, Trento, Italy.
- Rada Mihalcea and Dan Moldovan. 2001. Automatic generation of a coarse grained WordNet. In *Proceedings of NAACL Workshop on WordNet and Other Lexical Resources*, Pittsburgh, USA.
- Vivi Nastase and Michael Strube. 2013. Transforming Wikipedia into a large scale multilingual concept network. *Artificial Intelligence*, 194:62–85.
- Roberto Navigli and Simone Paolo Ponzetto. 2012a. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- Roberto Navigli and Simone Paolo Ponzetto. 2012b. Babel-Relate! a joint multilingual approach to computing semantic relatedness. In *Proceedings of AACL*, pages 108–114, Toronto, Canada.
- Roberto Navigli. 2006. Meaningful clustering of senses helps boost Word Sense Disambiguation performance. In *Proceedings of COLING-ACL*, pages 105–112, Sydney, Australia.
- Roberto Navigli. 2009. Word Sense Disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.
- Martha Palmer, Hoa Dang, and Christiane Fellbaum. 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13(2):137–163.
- Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. WordNet::Similarity - measuring the relatedness of concepts. In *Proceedings of AACL*, pages 144–152, San Jose, CA.
- Mohammad Taher Pilehvar and Roberto Navigli. 2013. Paving the way to a large-scale pseudosense-annotated dataset. In *Proceedings of NAACL-HLT*, pages 1100–1109, Atlanta, USA.
- Kira Radinsky, Eugene Agichtein, Evgeniy Gabilovich, and Shaul Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of WWW*, pages 337–346, Hyderabad, India.
- Daniel Ramage, Anna N. Rafferty, and Christopher D. Manning. 2009. Random walks for text semantic similarity. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 23–31, Suntec, Singapore.
- Reinhard Rapp. 2003. Word sense discovery based on sense descriptor dissimilarity. In *Proceedings of the Ninth Machine Translation Summit*, pages 315–322, New Orleans, LA.
- Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of IJCAI*, pages 448–453, Montreal, Canada.
- Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.
- Gerard Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.
- Rion Snow, Sushant Prakash, Daniel Jurafsky, and Andrew Y. Ng. 2007. Learning to merge word senses. In *EMNLP-CoNLL*, pages 1005–1014, Prague, Czech Republic.
- Mark Steyvers and Tom Griffiths. 2007. Probabilistic topic models. *Handbook of Latent Semantic Analysis*, 427(7):424–440.
- Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from Web collections. *Computational Linguistics*, 37(2):351–383.
- Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Peter D. Turney, Michael L. Littman, Jeffrey Bigham, and Victor Shnayder. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. In *Proceedings of RANLP*, pages 482–489, Borovets, Bulgaria.
- Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. 2012. Takelab: Systems for measuring semantic text similarity. In *Proceedings of SemEval-2012*, pages 441–448, Montreal, Canada.
- Michael J. Wise. 1993. String similarity via greedy string tiling and running Karp-Rabin matching. In *Department of Computer Science Technical Report*, Sydney.
- David Yarowsky. 1995. Unsupervised Word Sense Disambiguation rivaling supervised methods. In *Proceedings of ACL*, pages 189–196, Cambridge, Massachusetts.
- Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Using Wiktionary for computing semantic relatedness. In *Proceedings of AACL*, pages 861–866, Chicago, Illinois.

# Linking and Extending an Open Multilingual Wordnet

**Francis Bond**

Linguistics and Multilingual Studies  
Nanyang Technological University  
bond@ieee.org

**Ryan Foster**

Great Achievement Press  
ryan\_foster@gapbks.com

## Abstract

We create an open multilingual wordnet with large wordnets for over 26 languages and smaller ones for 57 languages. It is made by combining wordnets with open licences, data from Wiktionary and the Unicode Common Locale Data Repository. Overall there are over 2 million senses for over 100 thousand concepts, linking over 1.4 million words in hundreds of languages.

## 1 Introduction

We wish to create a lexicon covering as many languages as possible, with as much useful information as possible. Generally, language resources, to be useful, must be both **accessible** (legal to use) and **usable** (of sufficient quality, size and with a documented interface) (Ishida, 2006). We address both of these concerns in this paper.

One of the many attractions of the semantic network WordNet (Fellbaum, 1998), is that there are numerous wordnets being built for different languages. There are, in addition, many projects for groups of languages: Euro WordNet (Vossen, 1998), BalkaNet (Tufiş et al., 2004), Asian Wordnet (Charoenporn et al., 2008) and more. Although there are over 60 languages for which wordnets exist in some state of development (Fellbaum and Vossen, 2012, 316), less than half of these have released any data, and for those that have, the data is often not freely accessible (Bond and Paik, 2012). For those wordnets that are available, they are of widely varying size and quality, both in terms of accuracy and richness. Further, there is very little standardization in terms of format, what information is included, or license.

The goal of the research outlined in this paper is to make it possible for a researcher interested in working on the lexical semantics of a language or

languages to be able to access wordnets for those languages with a minimum of legal and technical barriers. In practice this means making it possible to access multiple wordnets with a common interface. We also use sources of semi-structured data that have minimal legal restrictions to automatically extend existing freely available wordnets and to create additional wordnets which can be added to our open wordnet grid.

Previous studies have leveraged multiple wordnets and Wiktionary (Wikimedia, 2013) to extend existing wordnets or create new ones (de Melo and Weikum, 2009; Hanoka and Sagot, 2012). These studies passed over the valuable sense groupings of translations within Wiktionary and merely used Wiktionary as a source of translations that were not disambiguated according to sense. The present study built and extended wordnets by directly linking Wiktionary senses to WordNet senses.

Meyer and Gurevych (2011) demonstrated the ability to automatically identify many matching senses in Wiktionary and WordNet based on the similarity of monolingual features. Our study combines monolingual features with the disambiguating power of multiple languages. In addition to differences in linking methodology, our project gives special attention to ensuring the maximum re-usability and accessibility of the data and software released.

Other large scale multilingual lexicons have been made by linking wordnet to Wikipedia (Wikipedia, 2013; de Melo and Weikum, 2010; Navigli and Ponzetto, 2012). Our approach is complementary to these: in general Wikipedia has more entities than classes, while Wiktionary has more classes.

In Section 2 we discuss linking freely available wordnets to form a single multilingual semantic network. In Section 3 we extend the wordnets with data from two sources. We show the results in Section 4 and then discuss them and outline future

work in Section 5.

## 2 Linking Multiple Wordnets

In order to make the data from existing wordnet projects more **accessible**, we have built a simple database with information from those wordnets with licenses that allow redistribution of the data. These wordnets, their licenses and recent activity are summarized in Table 1 (sizes for most of them are shown in Table 2).<sup>1</sup>

Wordnet Project	Lng	Licence	Type
Albanet <sup>o</sup>	als	CC BY	a
Arabic WordNet	arb	CC BY-SA	s
DanNet	dan	wordnet	a
Princeton WordNet <sup>u</sup>	eng	wordnet	a
Persian Wordnet	fas	free to use	u
FinnWordNet <sup>u</sup>	fin	CC BY	a
WOLF <sup>u</sup>	fra	CeCILL-C	s
Hebrew Wordnet <sup>o</sup>	heb	wordnet	s
MultiWordNet <sup>o</sup>	ita	CC BY	a
Japanese Wordnet <sup>u</sup>	jpn	wordnet	a
Multilingual	cat	CC BY	a
Central	eus	CC BY-NC-SA	n
Repository <sup>o,u</sup>	glg	CC BY	a
	spa	CC BY	a
Wordnet Bahasa <sup>u</sup>	ind	MIT	a
	zsm	MIT	a
Norwegian Wordnet <sup>o</sup>	nno	wordnet	a
	nob	wordnet	a
plWordNet <sup>o,u</sup>	pol	wordnet	a
OpenWN-PT <sup>u</sup>	por	CC BY-SA	s
Thai Wordnet	tha	wordnet	a

<sup>o</sup> Re-released under an open license in 2012

<sup>u</sup> Updated in 2012

Type: **u** Unrestricted; **a** Attribution; **s** Share-alike;

**n** Non-commercial

URL: <http://casta-net.jp/~kuribayashi/multi/>

Table 1: Linked Open Wordnets

The first wordnet developed is the Princeton WordNet (PWN: Fellbaum, 1998). It is a large lexical database of English. Open class words (nouns, verbs, adjectives and adverbs) are grouped into concepts represented by sets of synonyms (synsets). Synsets are linked by semantic relations such as hyponymy and meronymy. PWN is released under an open license (allowing one to use, copy, modify and distribute it so long as you properly acknowledge the copyright).

The majority of freely available wordnets take the basic structure of the PWN and add new lemmas (words) to the existing synsets: the **extend** model (Vossen, 2005). For example,  $dog_{n:1}$  is linked to the lemmas *chien* in French, *anjing* in Malay, and so on. It is widely realized that this

<sup>1</sup>We have now added Mandarin Chinese.

model is imperfect as different languages lexicalize different concepts and link them in different ways (Fellbaum and Vossen, 2012). Nevertheless, many projects have found that the overall structure of PWN serves as a useful scaffold. The fact that, for example, a  $dog_{n:1}$  is an  $animal_{n:1}$  is language independent.

In theory, such wordnets can easily be combined into a single resource by using the PWN synsets as pivots. All languages are linked through the English wordnet. Because they are linked at the synset level, the problem of ambiguity one gets when linking bilingual dictionaries through a common language is resolved: we are linking senses to senses.

In practice, linking a new language's wordnet into the grid could be problematic for three reasons. The first problem was that the wordnets were linked to various versions of the Princeton WordNet. In order to combine them into a single multilingual structure, we had to map to a common version. The second problem was the incredible variety of formats that the wordnets are distributed in. Almost every project uses a different format. Even different versions of the same project often had slightly different formats. The final problem was legal: not all wordnets have been released under licenses that allow reuse.

The first problem can largely be overcome using the mapping scripts from Daude et al. (2003). Mapping introduces some distortions, in particular, when a synset is split, we chose to only map the translations to the most probable mapping, so some new synsets will have no translations.

The second problem we are currently solving through brute force, writing a new script for every new project we add. We make these scripts, along with the reformatted wordnets, freely available for download. Any problems or bugs found when converting the wordnets have been reported back to the original projects, with many of them fixed in newer releases. We consider this feedback to be an important part of our work: it means that other researchers and users do not have to suffer from the same problems and it encourages projects to release updates.

The third, legal, problem is being solved by an ongoing campaign to encourage projects to (re-)release their data under open licenses. Since Bond and Paik (2012) surveyed wordnet licenses in 2011, six projects have newly released data un-

der open licenses and eight projects have updated their data.

Our combined wordnet includes English (Fellbaum, 1998); Albanian (Ruci, 2008); Arabic (Black et al., 2006); Chinese (Huang et al., 2010); Danish (Pedersen et al., 2009); Finnish (Lindén and Carlson., 2010); French (Sagot and Fišer, 2008); Hebrew (Ordan and Wintner, 2007); Indonesian and Malaysian (Nurril Hirfana et al., 2011); Italian (Pianta et al., 2002); Japanese (Isahara et al., 2008); Norwegian (Bokmål and Nynorsk: Lars Nygaard 2012, p.c.); Persian (Montazery and Faili, 2010); Portuguese (de Paiva and Rademaker, 2012); Polish (Piasecki et al., 2009); Thai (Thoongsup et al., 2009) and Basque, Catalan, Galician and Spanish from the Multilingual Common Repository (Gonzalez-Agirre et al., 2012).

On our server, the wordnets are all in a shared `sqlite` database using the schema produced by the Japanese WordNet project (Isahara et al., 2008). The database is based on the logical structure of the Princeton WordNet, with an additional language attribute for lemmas, examples, definitions and senses. It is a single open multilingual resource. When we redistribute the data, each project's data is made available separately, with a common format, but separate licenses.

The Scandinavian and Polish wordnets are based on the **merge** approach, where independent language specific structures are built and then some synsets linked to PWN. Typically only a small subset will be linked (due more to resource limitations than semantic incompatibility).

## 2.1 Core Concepts

Boyd-Graber et al. (2006) created a list of 5,000 **core** word senses in Princeton WordNet which represent approximately the 5,000 most frequently used word senses.<sup>2</sup> We use this list to evaluate the coverage of the wordnets: do they contain words for the most common concepts? As a very rough measure of useful coverage, we report the percentage of synsets covered from this core list. Because the list is based on English data, it is of course not a perfect measure for other languages and cultures. Note that some wordnet projects have deliberately targeted the core concepts, which of course boosts their coverage scores.

<sup>2</sup>The original list is here from <http://wordnetcode.princeton.edu/standoff-files/core-wordnet.txt>; we converted it to `wn30` synsets.

## 2.2 License Types

The licenses fall into four broad categories: **(u)** completely unrestricted, **(a)** attribution required, **(s)** share alike, and **(n)** non-commercial. The first category includes any work that is in the public domain or that the author has released without any restrictions. The second category allows anyone to use, adapt, improve, and redistribute the work as long as one attributes the work in the manner specified by the copyright holder (without suggesting an endorsement). The WordNet, MIT, and CC BY licenses are all in this category. The third category allows anyone to adapt and improve the licensed work and redistribute it, but the redistributed work must be released under the same license. The CC BY-SA, GPL, GFDL, and CeCILL-C licenses are of this type. Because derivative works can only be redistributed under the same license, works licensed under any two of these licenses cannot be combined with each other and legally redistributed. In general, a work formed from the combination of works in category **(u)** and **(a)** with a work in category **(s)** will be subject to the more restrictive terms of the the share alike license. However, the GPL, GFDL and CeCILL-C are incompatible with CC BY.<sup>3</sup> The fourth type of license further forbids the commercial use of a work. The CC BY-NC and the CC BY-NC-SA licenses are in this category, they are also incompatible with licenses in category **(s)**.

Releasing a work under the more restrictive licenses in categories **(s)** and **(n)** above substantially limit and complicate the ability to extend and combine a work into other useful forms. By maintaining a separation of databases released under incompatible licenses, we avoid any possible legal problems. Due to license incompatibilities, it is impossible to release a single database with all the wordnets, even though individually they are redistributable. We can currently combine those with licenses in groups **(u)** and **(a)** and the CC BY-SA wordnets (now everything except French and Basque).

## 3 Extending with non-wordnet data

We looked at two sources for automatically adding new entries. The Unicode Common Locale Data Repository (CLDR) has reliable information on languages, territories and dates. Wiktionary is a

<sup>3</sup><http://www.gnu.org/licenses/license-list.html#ccby>

general purpose lexicon with much more information for many words.

### 3.1 Unicode Common Locale Data Repository (CLDR)

We added information on languages, territories and dates from the Unicode Common Locale Data Repository (CLDR).<sup>4</sup> This is a collection of data maintained by the Unicode Consortium to support software internationalization and localization with locale information on formatting dates, numbers, currencies, times, and time zones, as well as help for choosing languages and countries by name. It has this data for over 194 languages. It is released under an open license that allows redistribution with proper attribution (Unicode, Inc., 2012).<sup>5</sup>

We found data for 122 languages. Most had around 550 senses (synsets and their lemmas): for example, for Portuguese: English<sub>n:1</sub> *inglês*. Some had only 40 or 50, such as Assamese, which only has the week days, month names and a few language names. The linked data was small enough to check by hand. When the original CLDR data is correct the data we generate should be correct.

The idea of using such data is not new. Quah et al. (2001) for example, use Linux locale data to extend a proprietary English-Malay lexicon. de Melo and Weikum (2009) also use this data (and data from a variety of other sources) to build an enhanced wordnet, in addition adding new synsets for concepts that are not in not wordnet. However, when they released the data as LEXVO (data about languages: CC BY-SA) and UWN (the universal multilingual wordnet: CC BY-NC-SA), they added additional license restrictions which complicate the reuse of the data and make it impossible to integrate the data back into the original wordnet project.

### 3.2 Wiktionary

Searches for a publicly-available source of Wiktionary in a preprocessed, machine-readable format did not turn up any sources that were recent and publicly-available.<sup>6</sup> Although there are sev-

eral freely-available software programs that are capable of parsing portions of the English Wiktionary, none of the programs that were evaluated appeared to extract the precise set of information desired for our task in an easy-to-use format. So the authors decided to build a custom parser capable of extracting the information needed for building open wordnets.

#### 3.2.1 Wiktionary Parser

Since each language edition of Wiktionary is formatted in a somewhat unique way, parsers must be tailored to recognize the structure and formatting of each edition on a case-by-case basis. The authors created a parser tailored to the English Wiktionary, although it can be extended to handle other language versions as well. We are releasing this code under the MIT license.<sup>7</sup>

The current version of the parser is capable of extracting headwords, parts of speech, definitions, synonyms and translations from the XML Wiktionary database dumps provided by the Wikimedia Foundation.<sup>8</sup> Within these large XML files, the main body of Wiktionary articles are stored in a Wikitext format, which is a semi-structured format. Although anyone can edit a Wiktionary page and use any style of formatting they desire, the community of users encourages adhering to established guidelines, which produces a format that is generally predictable.

Within the English Wiktionary, synonyms and translations are both grouped into sense groups that correspond with definitions in the main section. These sense groups are marked by a short text gloss (**short gloss**), which is usually an abbreviated version of one of the full definitions (**full definition**). The parser makes no attempt to match these short glosses with the full definitions. Data is simply extracted, cleaned, and then stored in a relational database or flat file.

Translations proved to be easy to extract due to the fairly consistent use of a specifically formatted translation template. These templates include a language code derived from ISO standards, the translation, and optional additional information such as gender, transliteration, script, and alternate forms. The parser extracts and retains all of this potentially valuable information.

Examples of translation templates:

<sup>7</sup>Available from the Open Multilingual Wordnet Page: <http://casta-net.jp/~kuribayashi/multi/>.

<sup>8</sup><http://dumps.wikimedia.org/>

<sup>4</sup><http://cldr.unicode.org/>

<sup>5</sup>With the extra requirement that “there is clear notice in each modified Data File or in the Software as well as in the documentation associated with the Data File(s) or Software that the data or software has been modified.”

<sup>6</sup>We later learned that McCrae et al. (2012) made a release of Wiktionary in the lemon format (<http://datahub.io/en/dataset/dbnary>). They did not, however, release the code they used to parse Wiktionary.

- Finnish:  $\{\{t+|fi|sanakirja\}\}$
- French:  $\{\{t+|fr|dictionnaire|m\}\}$

To enable later processing, it is necessary to tie synonyms and translations to their corresponding short gloss via a unique key. Most parsers simply use an automatically generated surrogate key or a key based on the ordered position of data within a Wiktionary article. Since Wiktionary is constantly changing, the side effect of this approach is that data extracted from a specific snapshot of the Wiktionary database can only be meaningfully used in connection with other data extracted by the same parser from the exact same snapshot. To overcome this, we use a unique key that can be recreated from the data itself, which we call the **defkey**. To generate this key, we concatenate the language code, headword, part of speech, and the short gloss and use the sha1 hash function (NIST, 2012) to create a unique 40-character hexadecimal string from the resulting text.

These defkeys are time and technology independent, so they allow the ability for researchers to efficiently share and compare results. Once a link is established between this defkey and a particular synset, translations added to Wiktionary at a later data can be automatically integrated into our multilingual wordnet. Conversely, if a Wiktionary contributor changes a short gloss, historical data connected to the old defkey is preserved while new data imported at a later time will not be incorrectly linked to an older definition.

Another feature of our parser is a feedback mode, which generates a report about poorly formatted data that was encountered. These automatically generated reports can be used to create a quality-enhancing feedback loop with Wiktionary.

### 3.2.2 Linking Senses

Meyer and Gurevych (2011) showed that automatic alignments between Wiktionary senses and PWN can be established with reasonable accuracy and recall by combining multiple text similarity scores to compare a bag of words based on several pieces of information linked to a WordNet sense with another bag of words obtained from a Wiktionary entry. In our study we evaluated the potential for aligning senses based on common translations in combination with monolingual similarity features.

In this study we used 20 of the wordnets de-

scribed in Section 2,<sup>9</sup> and the Wiktionary data obtained using the parser described in Section 3.2.1. Before searching for translation matches, we normalized the data to ensure the most accurate possible overlap count. First, article headwords were included as English translations of Wiktionary senses (along with synonyms). Then differences in language codes were rectified and translations containing symbolic characters or a mixture of roman and non-roman characters were marked to be ignored, save a few exceptions. This left approximately 1.4 million sense translations in 20 languages in our wordnet grid, and nearly 1.3 million Wiktionary translations in over 1,000 languages.

We then created a list of all possible alignments where at least one translation of a wordnet sense matched a translation of a Wiktionary sense. This represented a small percentage of the possible alignments, because definitions in Wiktionary that do not contain any translations were ignored in our study. Of more than 500,000 English definitions in Wiktionary, only about 130,000 presently have associated translations. The resulting graph contained over 700,000 possible sense alignments.

We calculated a number of similarity scores, the first two based on similarity in the number of lemmas, calculated using the Jaccard index:

$$\text{sim}_e(s_n, s_k) = \frac{|E(s_k) \cap E(s_n)|}{|E(s_k) \cup E(s_n)|} \quad (1)$$

$$\text{sim}_a(s_n, s_k) = \frac{|L(s_k) \cap L(s_n)|}{|L(s_k) \cup L(s_n)|} \quad (2)$$

Where  $s_k, s_n$  are concepts in Wiktionary and wordnet respectively,<sup>10</sup>  $E(s)$  is the set of English lemmas for sense  $s$  and  $L(s)$  is the set of lemmas in all languages.

As an initial pruning, we kept only matches where either:  $\text{sim}_a \geq 0.7$  **or** ( $\text{sim}_e \geq 0.5$  **and**  $\text{sim}_a \geq 0.5$ ) **or**, if  $(|L(s_k) \cap L(s_n)| > 5)$  then ( $\text{sim}_e \geq 0.5$  **and**  $\text{sim}_a \geq 0.45$ ). After applying these filters, approximately 220,000 alignment candidates remained.

We reviewed a random sample of 551 alignment candidates. Of these 136 were deemed correctly aligned. Another 48 we considered possibly close enough to produce valid translations for wordnet. All others were marked as incorrect alignments.

<sup>9</sup>We didn't use Chinese or Polish, as the wordnets were added after we had started the evaluation.

<sup>10</sup>Precisely, synsets in wordnet and senses in Wiktionary.

This development dataset was used to tune refined similarity scores.

$$\text{sim}_t(s_n, s_k) = \frac{|L(s_k) \cap L(s_n)|}{\sqrt{\alpha |L(s_k) \cup L(s_n)|}} \quad (3)$$

$$\text{sim}_d(s_n, s_k) = \frac{\text{BoW}(\text{wndef}) \cdot \text{BoW}(\text{wkdef})}{\|\text{BoW}(\text{wndef})\| \|\text{BoW}(\text{wkdef})\|} \quad (4)$$

$$\text{sim}_c(s_n, s_k) = \text{sim}_t + \beta \text{sim}_d \quad (5)$$

$\text{sim}_t$  gives higher weight to concepts that link through more lemmas, not just a higher proportion of lemmas.

$\text{sim}_d$  measures the similarity of the definitions in the two resources, using a cosine similarity score. We initially used the WordNet gloss and example sentence(s) for `wndef` and the short gloss from Wiktionary for `wkdef`. This improved the accuracy of the combined ranking score ( $\text{sim}_c$ ), but since many of the short glosses are only one or two words, the sparse input often produced a  $\text{sim}_d$  score of zero even when the candidate alignment was correct. To improve the accuracy of the  $\text{sim}_d$  component, we also added in the long definitions.

Short glosses were aligned with long definitions using a similar approach to McCrae et al. (2012). First we search for a match where the short gloss was a substring of the full definition. If that failed to produce a single possible alignment, we aligned the short gloss with the full definition that produced the greatest cosine similarity score. Finally, where the short definition was blank and only a long definition was present, we aligned the two. The results of this alignment were less than 90% accurate, so to offset the effects of this noise we included both the full definition and the short gloss in `wkdef`. For `wndef` we used the WordNet gloss, example sentence(s), and synonyms. Even though the linking of definitions within Wiktionary left much to be desired, the increased amount of text improved the accuracy of the definition based similarity component of our ranking score.

Our combined ranking score ( $\text{sim}_c$ ), based on both overlapping translations and a monolingual lexical similarity score, was able to outperform ranking based on either component in isolation. We expect that an improved alignment of short glosses to full definitions together with more accurate measures of lexical similarity such as described by Meyer and Gurevych (2011) would further improve the accuracy of a combined ranking score. We employed our combined ranking score first as a filter, where  $\text{sim}_c \geq \tau_c$ . The ranking score

is then used to select the best match among competing alignments. Alignments are based on the belief that a definition within Wiktionary should only map to a single WordNet synset (if any at all). In theory, each WordNet synset should represent a meaning distinguishable from all other synsets. Because Wiktionary is organized according to lemma first, and sense second, multiple definitions in separate articles often map to the same synset. For example *mortal* “A human; someone susceptible to death”, *individual* “A person considered alone ...”, and *person* “A single human being; an individual” all align with `someonen:1` (00007846-n). However, two distinct definitions within the same Wiktionary entry should not map to the same WordNet sense. When there are multiple possible alignments where only one can be valid,  $\text{sim}_c$  is used to determine the best match.

In addition to using the combined ranking score as a filter, we found that we could obtain a small additional increase in accuracy without reducing recall by also requiring  $\text{sim}_t \geq \tau_t$  or  $\text{sim}_d \geq \tau_d$ .

To determine ideal values for the weights and thresholds, we performed several grid searches. The parameters are interdependent and can produce reasonable results at a variety of points. Ideal values also depend on whether we wish to maximize accuracy or recall.  $\alpha$  is set at 3.2 in order to achieve an ideal target threshold of  $\tau_t = 1$ . We finally chose values of  $\beta = 0.7$  and  $\tau_c = 0.71$  which gave a reasonable balance between accuracy and recall.

## 4 Results and Evaluation

We give the data for the 26 wordnets with more than 10,000 synsets in Table 2. There are a further 57 with more than 1,000; 133 with more than 100, 200 with more than 10 and 645 with more than 1 (although most of the very small languages appear to be simple errors in the language code entered into Wiktionary). Individual totals are shown for synsets and senses from the original wordnets, the data extracted from Wiktionary, and the merged data of the wordnets, Wiktionary and CLDR. We do not show the CLDR data in the table as it is so small, generally 500-600 synsets for the top languages. Overall there are 2,040,805 senses for 117,659 concepts, using over 1,400,000 words in over 1,000 languages.

The smaller wordnets are not of much practical use, but can still serve as the core of new

ISO	Language	Projects			Wiktionary			Merged (+CLDR)		
		Synsets	Senses	Core	Synsets	Senses	Core	Synsets	Senses	Core
eng	English	117,659	206,978	100	35,400	49,951	75	117,661	213,538	100
fin	Finnish	116,763	189,227	100	21,516	31,154	65	116,830	199,435	100
tha	Thai	73,350	95,517	81	2,560	3,193	17	73,595	97,390	81
fra	French	59,091	102,671	92	20,449	27,150	63	61,258	109,643	95
jpn	Japanese	57,179	158,064	95	12,685	19,479	52	59,112	166,617	96
ind	Indonesian	52,006	142,488	99	2,390	2,810	17	52,154	143,755	99
cat	Catalan	45,826	70,622	81	8,626	10,251	36	48,007	74,806	84
spa	Spanish	38,512	57,764	76	18,281	25,310	60	47,737	74,848	86
por	Portuguese	41,810	68,285	79	12,331	16,178	53	43,870	74,151	84
zsm	Standard Malay	42,766	119,152	99	2,833	3,744	19	43,079	120,686	99
ita	Italian	34,728	60,561	83	14,605	18,710	53	38,938	68,827	87
eus	Basque	29,413	48,934	71	1,693	1,943	11	29,965	49,945	72
pol	Polish	14,008	21,001	30	10,888	13,431	46	20,975	30,943	55
glg	Galician	19,312	27,138	36	2,492	2,871	15	20,772	29,136	42
fas	Persian	17,759	30,461	41	4,229	5,443	26	20,766	35,318	55
rus	Russian	0	0	0	19,983	33,716	64	20,138	34,009	64
deu	German	0	0	0	19,675	29,616	64	19,857	29,884	64
cmn	Mandarin Chinese	4,913	8,069	28	12,130	19,079	49	15,490	27,113	60
arb	Standard Arabic	10,165	21,751	48	6,892	9,337	38	14,861	31,337	63
nld	Dutch	0	0	0	13,741	19,709	56	13,950	20,003	56
ces	Czech	0	0	0	12,802	15,493	54	13,030	15,813	54
swe	Swedish	0	0	0	12,000	16,226	51	12,221	16,512	51
ell	Modern Greek	0	0	0	10,308	13,071	44	10,549	13,472	44
dan	Danish	4,476	5,859	81	7,290	8,931	35	10,328	13,551	85
nob	Norwegian Bokmål	4,455	5,586	79	7,262	9,170	35	10,322	13,612	83
hun	Hungarian	0	0	0	9,964	12,699	45	10,213	13,029	45

**Core** shows the percentage coverage of the 5,000 core concepts.

Table 2: Merged Wordnets (with more than 10,000 entries)

projects. The bigger wordnets show the data from Wiktionary (and to a lesser extent CLDR) having only a small increase in the number of senses. The biggest change is for the medium size projects, such as Persian or Arabic, which end up with much better coverage of the most frequent core concepts. Major languages such as German or Russian, which currently do not have open wordnets get good coverage as well.

The size of the mapping table is the same as the number of English senses linked (49,951 senses). We evaluated a random sample of 160 alignments and found the accuracy to be 90% (Wiktionary sense maps to the best possible wordnet sense).

We then evaluated samples of the wordnet created from Wiktionary for several languages. For each language we choose 100 random senses, then checked them against existing wordnets.<sup>11</sup> For all unmatched entries, we then had them checked by native speakers. The results are given in Table 3. The sense accuracy is higher than the mapping accuracy: in general, entries with more translations are linked more accurately, thus raising the average precision. During the extraction and eval-

Language	% Matched	% Good
Chinese*	46	97
Serbo-Croatian*,**	0	91
Czech*	0	99
English	89	92
German*	19	85
Indonesian	69	97
Korean*	0	96
Japanese	56	90
Russian*	0	99
Average		94.0

Table 3: Precision of Wiktionary-based Wordnets

\* Not used to build the mapping from wordnet to Wiktionary.  
 \*\* We allow terms used in either Serbian or Croatian.

uation, we noticed several language specific features: for example, Serbo-Croatian had a mixture of Cyrillic and Latin entries. For languages where one script was clearly dominant, we kept only that, but really these decisions should be done for each language by a native speaker.

We make the data available in two ways. The first is a set of downloads. Each language has up to three files: the data from the wordnet project (if it exists), the data from the CLDR and the data from Wiktionary. They are kept separate in order

<sup>11</sup>For Chinese we use the wordnet from Xu et al. (2008), which is free for research but cannot be redistributed. For German we used Euro WordNet (Vossen, 1998).

to keep the licenses as free as possible. The second is as two on-line searches: one using only the data from the projects, and one with all the data combined. The combination is done by simple union.<sup>12</sup> We maintain this separation as we cannot guarantee the quality of the automatically extracted data. Because the raw data is there it is possible to combine them in other ways. The simple structure is easy to manipulate, and there is code to use this style of data with the popular tool kit NLTK (Bird et al., 2010).

## 5 Discussion and Future Work

We have created a large open wordnet of high quality (85%–99% measured on senses). Twenty six languages have more than 10,000 concepts covered, with 42–100% coverage of the most common core concepts. The data is easily downloadable with minimal restrictions. The overall accuracy is estimated at over 94%, as most of the original wordnets are hand verified (and so should be 100% accurate). The high accuracy is largely thanks to the disambiguating power of the multiple translations, made possible by the many open wordnets we have access to.

Because we link senses between wordnet and Wiktionary and then use the translations of the sense, manually validating this mapping will improve the entries in multiple languages simultaneously. As the Wiktionary-wordnet alignment mapping is linked to persistent keys it will remain useful even as the resources change. Further, it can be used to identify and add missing senses to wordnet: unmapped Wiktionary entries are candidates for new concepts.

The Universal Wordnet (UWN: de Melo and Weikum, 2009) brings in data from even more resources, and combines them to make a larger resource, choosing parameters with slightly lower precision (just under 90%). It is further linked to Wikipedia, adding many named entities. We expect that our work is complementary. Because we use a different approach, it would be possible to merge the two if the licenses allowed us to. However, since the CC BY-SA and CC-BY-NC-SA licenses are mutually exclusive, the two works cannot be combined and rereleased unless relevant parties can relicense the works. There is no easy way to improve UWN beyond checking each and every entry, which is expensive. An ad-

vantage of our approach, noted above, is that we can validate the sense matches for English and the accuracy percolates down to all the languages.

Integrating data from the most recent version of Wiktionary can be done simply and takes a few hours. It is therefore feasible to update the downloadable data regularly. Improvements in either the wordnet projects or Wiktionary (or both) can also result in improved mappings. We further hope to take advantage of ongoing initiatives in the global wordnet grid to add new concepts not in the Princeton WordNet, so that we can expand beyond an English-centered world view.

By making the data from multiple sources easily available with minimal restrictions, we hope that it will be easier to do research that exploits lexical semantics. In particular, we make the data easily accessible to the original wordnet projects, some of whom have already started to merge it into their own resources. We cannot check the accuracy of data in all languages, nor, for example, check that synsets have the most appropriate lemmas associated with them. Many languages have their own orthographic issues (for example a choice of scripts, or the choice to include vowels or not). Our automatic extraction does not deal with these issues at all. This kind of language specific quality control is best done by the individual wordnet projects.

We also consider it important to keep feeding data back to the individual wordnet projects, as much of the innovative research comes from them: the class/instance distinction from PWN; the distinction between rigid and non-rigid synsets from the Kyoto Project; domain mappings from the MultiWordNet (Pianta et al., 2002); representing orthographic variation from the Japanese Wordnet (Kuroda et al., 2011); combining close languages from the Wordnet Bahasa (Nurril Hirfana et al., 2011); and so on. For all of these reasons, we do not consider automatic extraction from/linking to Wiktionary a substitute for building languages specific wordnets.

Further work that this data should allow us to do include: automatically producing a list of bad data found in Wiktionary that can be used by Wiktionary editors to correct errors; and finding gaps in wordnet by identifying senses in Wiktionary that have a large number of translations, but fail to have any significant alignment with existing wordnet synsets.

<sup>12</sup><http://casta-net.jp/~kuribayashi/multi/>

We currently only link through the English Wiktionary and its translations. It should be possible to expand the multilingual wordnet in the same way using Wiktionaries in other languages, which we would expect to improve coverage.

Finally, Wiktionary contains a lot of useful information we are not currently using (information on gender, transliterations, pronunciations, alternative spellings and so forth). We can also think of the aligned definitions as a paraphrase corpus for English.

We have devoted more space than is usual for a computational linguistics paper to issues of licensing and sustainability. This is deliberate: we feel papers about lexical resources should be clear about licensing, and that it should be considered early on when creating new resources. There are strong arguments that open data leads to better science (Pederson, 2008), and it has been shown that open resources are cited more (Bond and Paik, 2012). In addition, how to maintain resources over time is a major unsolved problem. We consider it important that our wordnet is not just large and accurate but also maintainable and as accessible as possible.

## 6 Conclusions

We have created an open multilingual wordnet with over 26 languages. It is made by combining wordnets with open licences, data from the Unicode Common Locale Data Repository and Wiktionary. Overall there are over 2 million senses for 117,659 concepts, using over 1.4 million words in hundreds of languages.

## Acknowledgments

We would like to thank the following for their help with the evaluation: Le Tuan Anh, František Kratochvíl, Kyonghee Paik, Zina Pozen, Melanie Siegel, Stefanie Stadler, Bilyana Shuman, Liling Tan and Muhammad Zulhelmy bin Mohd Rosman.

## References

Stephen Bird, Ewan Klein, and Edward Loper. 2010. *Nyumon Shizen Gengo Shori [Introduction to Natural Language Processing]*. O'Reilly. (translated by Hagiwara, Nakamura and Mizuno).

W. Black, S. Elkateb, H. Rodriguez, M. Alkhalifa, P. Vossen, A. Pease, M. Bertran, and C. Fell-

baum. 2006. The Arabic wordnet project. In *Proceedings of LREC 2006*.

Francis Bond and Kyonghee Paik. 2012. A survey of wordnets and their licenses. In *Proceedings of the 6th Global WordNet Conference (GWC 2012)*. Matsue. 64–71.

Jordan Boyd-Graber, Christiane Fellbaum, Daniel Osherson, and Robert Schapire. 2006. Adding dense, weighted connections to WordNet. In *Proceedings of the Third Global WordNet Meeting*. Jeju.

Thatsanee Charoenporn, Virach Sornlerlamvanich, Chumpol Mokrat, and Hitoshi Isahara. 2008. Semi-automatic compilation of Asian WordNet. In *14th Annual Meeting of the Association for Natural Language Processing*, pages 1041–1044. Tokyo.

Jordi Daude, Lluís Padro, and German Rigau. 2003. Validation and tuning of Wordnet mapping techniques. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP'03)*. Borovets, Bulgaria.

Gerard de Melo and Gerhard Weikum. 2009. Towards a universal wordnet by learning from combined evidence. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 513–522. ACM, New York, NY, USA.

Gerard de Melo and Gerhard Weikum. 2010. Towards universal multilingual knowledge bases. In Pushpak Bhattacharyya, Christiane Fellbaum, and Piek Vossen, editors, *Principles, Construction, and Applications of Multilingual Wordnets. Proceedings of the 5th Global WordNet Conference (GWC 2010)*, pages 149–156. Narosa Publishing, New Delhi, India.

Valeria de Paiva and Alexandre Rademaker. 2012. Revisiting a Brazilian wordnet. In *Proceedings of the 6th Global WordNet Conference (GWC 2012)*. Matsue.

Christiane Fellbaum and Piek Vossen. 2012. Challenges for a multilingual wordnet. *Language Resources and Evaluation*, 46(2):313–326. Doi=10.1007/s10579-012-9186-z.

Christine Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Aitor Gonzalez-Agirre, Egoitz Laparra, and German Rigau. 2012. Multilingual central repos-

- itory version 3.0: upgrading a very large lexical knowledge base. In *Proceedings of the 6th Global WordNet Conference (GWC 2012)*. Matsue.
- Valérie Hanoka and Benoît Sagot. 2012. Wordnet creation and extension made simple: A multilingual lexicon-based approach using wiki resources. In *Proceedings of LREC 2012*. Istanbul.
- Chu-Ren Huang, Shu-Kai Hsieh, Jia-Fei Hong, Yun-Zhu Chen, I-Li Su, Yong-Xiang Chen, and Sheng-Wei Huang. 2010. Chinese wordnet: Design and implementation of a cross-lingual knowledge processing infrastructure. *Journal of Chinese Information Processing*, 24(2):14–23. (in Chinese).
- Hitoshi Isahara, Francis Bond, Kiyotaka Uchi-moto, Masao Utiyama, and Kyoko Kanzaki. 2008. Development of the Japanese WordNet. In *Sixth International conference on Language Resources and Evaluation (LREC 2008)*. Marrakech.
- Toru Ishida. 2006. Language grid: An infrastructure for intercultural collaboration. In *IEEE/IPSJ Symposium on Applications and the Internet (SAINT-06)*, pages 96–100. URL <http://langrid.nict.go.jp/file/langrid20060211.pdf>, (keynote address).
- Kow Kuroda, Takayuki Kuribayashi, Francis Bond, Kyoko Kanzaki, and Hitoshi Isahara. 2011. Orthographic variants and multilingual sense tagging with the Japanese WordNet. In *17th Annual Meeting of the Association for Natural Language Processing*, pages A4–1. Toyohashi.
- Krister Lindén and Lauri Carlson. 2010. Finnwordnet — wordnet påfinska via översättning. *LexicoNordica — Nordic Journal of Lexicography*, 17:119–140. In Swedish with an English abstract.
- John McCrae, Philipp Cimiano, and Elena Montiel-Ponsoda. 2012. Integrating wordnet and wiktionary with lemon. In Christian Chiarcos, Sebastian Nordhoff, and Sebastian Hellman, editors, *Linked Data in Linguistics*. Springer.
- Christian M. Meyer and Iryna Gurevych. 2011. What psycholinguists know about chemistry: Aligning wiktionary and wordnet for increased domain coverage. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 883–892.
- Nurri Hirfana Mohamed Noor, Suerya Sapuan, and Francis Bond. 2011. Creating the open Wordnet Bahasa. In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation (PACLIC 25)*, pages 258–267. Singapore.
- Mortaza Montazery and Hesham Faili. 2010. Automatic Persian wordnet construction. In *23rd International conference on computational linguistics*, pages 846–850.
- Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.
- NIST. 2012. Secure hash standard (shs). Fips pub 180-4, National Institute of Standards and Technology.
- Noam Ordan and Shuly Wintner. 2007. Hebrew wordnet: a test case of aligning lexical databases across languages. *International Journal of Translation*, 19(1):39–58.
- B.S Pedersen, S. Nimb, J. Asmussen, N. Sørensen, L. Trap-Jensen, and H. Lorentzen. 2009. DanNet — the challenge of compiling a wordnet for Danish by reusing a monolingual dictionary. *Language Resources and Evaluation*.
- Ted Pederson. 2008. Empiricism is not a matter of faith. *Computational Linguistics*, 34(3):465–470.
- Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. 2002. Multiwordnet: Developing an aligned multilingual database. In *Proceedings of the First International Conference on Global WordNet*, pages 293–302. Mysore, India.
- Maciej Piasecki, Stan Szpakowicz, and Bartosz Broda. 2009. *A Wordnet from the Ground Up*. Wroclaw University of Technology Press. URL [http://www.plwordnet.pwr.wroc.pl/main/content/files/publications/A\\_Wordnet\\_from\\_the\\_Ground\\_Up.pdf](http://www.plwordnet.pwr.wroc.pl/main/content/files/publications/A_Wordnet_from_the_Ground_Up.pdf), (ISBN 978-83-7493-476-3).
- Chiew Kin Quah, Francis Bond, and Takefumi Yamazaki. 2001. Design and construction of a machine-tractable Malay-English lexicon.

- In *Asialex 2001 Proceedings*, pages 200–205. Seoul.
- Ervin Ruci. 2008. On the current state of Albanet and related applications. Technical report, University of Vlora. (<http://fjalnet.com/technicalreportalbanet.pdf>).
- Benoît Sagot and Darja Fišer. 2008. Building a free French wordnet from multilingual resources. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco.
- Sareewan Thoongsup, Thatsanee Charoenporn, Kergrit Robkop, Tan Sinthurahat, Chumpol Mokarat, Virach Sornlertlamvanich, and Hitoshi Isahara. 2009. Thai wordnet construction. In *Proceedings of The 7th Workshop on Asian Language Resources (ALR7), Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics (ACL) and the 4th International Joint Conference on Natural Language Processing (IJCNLP)*,. Suntec, Singapore.
- Dan Tufiş, Dan Cristea, and Sofia Stamou. 2004. BalkaNet: Aims, methods, results and perspectives. a general overview. *Romanian Journal of Information Science and Technology*, 7(1–2):9–34.
- Unicode, Inc. 2012. Unicode, Inc. license agreement - data files and software. <http://www.unicode.org/copyright.html>.
- Piek Vossen, editor. 1998. *Euro WordNet*. Kluwer.
- Piek Vossen. 2005. Building wordnets. <http://www.globalwordnet.org/gwa/BuildingWordnets.ppt>.
- Wikimedia. 2013. List of wiktionaries. <http://meta.wikimedia.org/w/index.php?title=Wiktioary&oldid=4729333>. (accessed on 2013-02-14).
- Wikipedia. 2013. Wikipedia — wikipedia, the free encyclopedia. URL <http://en.wikipedia.org/w/index.php?title=Wikipedia&oldid=552515903>, [Online; accessed 30-April-2013].
- Renjie Xu, Zhiqiang Gao, Yuzhong Qu, and Zhisheng Huang. 2008. An integrated approach for automatic construction of bilingual Chinese-English WordNet. In *3rd Asian Semantic Web Conference (ASWC 2008)*, pages 302–341.

# FrameNet on the Way to Babel: Creating a Bilingual FrameNet Using Wiktionary as Interlingual Connection

Silvana Hartmann<sup>†</sup> and Iryna Gurevych<sup>†‡</sup>

<sup>†</sup> Ubiquitous Knowledge Processing Lap (UKP-TUDA)

Department of Computer Science, Technische Universität Darmstadt

<sup>‡</sup> Ubiquitous Knowledge Processing Lap (UKP-DIPF)

German Institute for Educational Research and Educational Information

[www.ukp.tu-darmstadt.de](http://www.ukp.tu-darmstadt.de)

## Abstract

We present a new bilingual FrameNet lexicon for English and German. It is created through a simple, but powerful approach to construct a FrameNet in any language using Wiktionary as an interlingual representation. Our approach is based on a sense alignment of FrameNet and Wiktionary, and subsequent translation disambiguation into the target language. We perform a detailed evaluation of the created resource and a discussion of Wiktionary as an interlingual connection for the cross-language transfer of lexical-semantic resources. The created resource is publicly available at <http://www.ukp.tu-darmstadt.de/fnwkde/>.

## 1 Introduction

FrameNet is a valuable resource for natural language processing (NLP): semantic role labeling (SRL) systems based on FrameNet provide semantic analysis for NLP applications, such as question answering (Narayanan and Harabagiu, 2004; Shi and Mihalcea, 2005) and information extraction (Mohit and Narayanan, 2003). However, their wide deployment has been prohibited by the poor coverage and limited availability of a similar resource in many languages.

Expert-built lexical-semantic resources are expensive to create. Previous cross-lingual transfer of FrameNet used corpus-based approaches, or resource alignment with multilingual expert-built resources, such as EuroWordNet. The latter indirectly also suffers from the high cost and constrained coverage of expert-built resources.

Recently, collaboratively created resources have been investigated for the multilingual extension of resources in NLP, beginning with Wikipedia (Navigli and Ponzetto, 2010). They rely on the so-called “Wisdom of the Crowds”, contributions by

a large number of volunteers, which results in a continuously updated high-quality resource available in hundreds of languages. Due to the encyclopedic nature of Wikipedia, previous work focused on encyclopedic information for Wikipedia entries, i.e., almost exclusively on nouns.

This is not enough for resources like FrameNet. Such resources need *lexical-semantic information* on *various POS*. For FrameNet, information on the predicates associated with a semantic frame – mostly verbs, nouns, and adjectives – is crucial, for instance gloss or syntactic subcategorization.

A solution for the problem of multilingual extension of lexical semantic resources is to use Wiktionary, a collaboratively created dictionary, as connection between languages. It provides high-quality lexical information on all POS, for instance glosses, sense relations, syntactic subcategorization, etc. Like Wikipedia, it is continuously extended and contains translations to hundreds of languages, including low-resource ones. To our knowledge, Wiktionary has not been evaluated as an interlingual index for the cross-lingual extension of lexical-semantic resources.

In this paper, we present a novel method for the creation of bilingual FrameNet lexicons based on an alignment to Wiktionary. We demonstrate our method on the language pair English-German and present the resulting resources, a lemma-based multilingual and a sense-disambiguated German-English FrameNet lexicon.

The understanding of lexical-semantic resources and their combinations, e.g., how alignment algorithms can be adapted to individual resource pairs and different POS, is essential for their effective use in NLP and a prerequisite for later in-task evaluation and application. To enhance this understanding for the presented resource pair, we perform a detailed analysis of the created resource and compare it to existing FrameNet resources for German.

The contributions of our work are the following: (1) We create a novel sense alignment between FrameNet and the English Wiktionary. It results in a multilingual FrameNet FNWKxx, which links FrameNet senses to lemmas in 280 languages. (2) We create a sense-disambiguated English-German FrameNet lexicon FNWKde based on FNWKxx and translation disambiguation on the German Wiktionary.<sup>1</sup> (3) We analyze the two resources and outline further steps for creating a multilingual FrameNet.

This is a major step towards the vision of this paper: a simple, but powerful approach to partially construct a FrameNet in any language using Wiktionary as an interlingual representation.

## 2 Resource Overview

**FrameNet** (Baker et al., 1998) is an expert-built lexical-semantic resource incorporating the theory of frame-semantics (Fillmore, 1976). It groups word senses in frames that represent particular situations. Thus, the verb *complete* and the noun *completion* belong to the *Activity finish* frame. The participants of these situations, typically realized as syntactic arguments, are the *semantic roles* of the frame, for instance the *Agent* performing an activity, or the *Activity* itself. FrameNet release 1.5 contains 1,015 frames, and 11,942 word senses. Corpus texts annotated with frames and their roles have been used to train automatic SRL systems.

**Wiktionary** is a collaboratively created dictionary available in over 500 language editions. It is continuously extended and revised by a community of volunteer editors. The English language edition contains over 500,000 word senses.<sup>2</sup>

Wiktionary is organized like a traditional dictionary in lexical entries and word senses. For the word senses, definitions and example sentences, as well as other lexical information, such as register (e.g., *colloquial*), phonetic transcription, inflection may be available, including language-specific types of information. Senses also provide translations to other languages. These are connected to lexical entries in the respective language editions via hyperlinks. This allows us to use Wiktionary as an interlingual connection between multiple languages.

<sup>1</sup>The xx in FNWKxx stands for all the languages in the resource. After translation disambiguation in a specific language, xx is replaced by the corresponding language code.

<sup>2</sup>as of May 2013, see <http://en.wiktionary.org/wiki/Wiktionary:Statistics>.

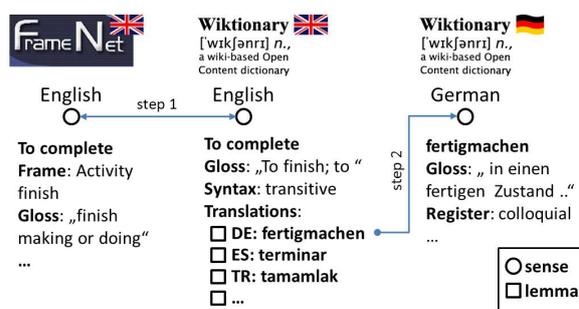


Figure 1: Method overview.

The quality of Wiktionary has been confirmed by Meyer and Gurevych (2012b) who also give an overview on the usage of Wiktionary in NLP applications such as speech synthesis.

## 3 Method Overview

Our method consists of two steps visualized in Fig. 1. In the first step, we create a novel sense alignment between FrameNet and the English Wiktionary following Niemann and Gurevych (2011). Thus, the FrameNet sense of *to complete* with frame *Activity finish* is assigned to the sense of *to complete* in Wiktionary meaning *to finish*.

This step establishes Wiktionary as an interlingual index between FrameNet senses and lemmas in many languages, and builds the foundation for the bilingual FrameNet extension.

It results in a basic multilingual FrameNet lexicon FNWKxx with translations to lemmas in 283 languages. An example: by aligning the FrameNet sense of the verb *complete* with gloss *to finish* with the corresponding English Wiktionary sense, we collect 39 translations to 22 languages, e.g., the German *fertigmachen* and the Spanish *terminar*.

The second step is the disambiguation of the translated lemmas with respect to the target language Wiktionary in order to retrieve the linguistic information of the corresponding word sense in the target language Wiktionary (Meyer and Gurevych, 2012a). We evaluate this step for English and German and create the bilingual FrameNet lexicon FNWKde. For the example sense of *complete*, we extract lexical information for the word sense of its German translation *fertigmachen*, for instance a German gloss, an example sentence, register information (*colloquial*), and synonyms, e.g., *beenden*. As a side-benefit of our method, we also extend the English FrameNet by the linguistic information in Wiktionary.

## 4 Related Work

### 4.1 Creating FrameNets in New Languages

There are two main lines of research in bootstrapping a FrameNet for languages other than English.

The first, corpus-based approach is to automatically extract word senses in the target language based on parallel corpora and frame annotations in the source language. In this vein, Padó and Lapata (2005) propose a cross-lingual FrameNet extension to German and French; Johansson and Nugues (2005) and Johansson and Nugues (2006) do this for Spanish and Swedish, and Basili et al. (2009) for Italian.

Padó and Lapata (2005) observe that their approach suffers from polysemy errors, because lemmas in the source language need to be disambiguated with respect to all the frames they evoke. To alleviate this problem, they use a disambiguation approach based on the most frequent frame; Basili et al. (2009) use distributional methods for frame disambiguation. Our approach is based on sense alignments and therefore explicitly aims to avoid such errors.

The second line of work is resource-based: FrameNet is aligned to multilingual resources in order to extract senses in the target language. Using monolingual resources, this approach has also been employed to extend FrameNet coverage for English (Shi and Mihalcea, 2005; Johansson and Nugues, 2007; Ferrandez et al., 2010).

De Cao et al. (2008) map FrameNet frames to WordNet synsets based on the embedding of FrameNet lemmas in WordNet. They use MultiWordNet, an English-Italian wordnet, to induce an Italian FrameNet lexicon with 15,000 entries.

To create *MapNet*, Tonelli and Pianta (2009) align FrameNet senses with WordNet synsets by exploiting the textual similarity of their glosses. The similarity measure is based on stem overlap of the candidates' glosses expanded by WordNet domains, the WordNet synset, and the set of senses for a FrameNet frame. In Tonelli and Pighin (2009), they use these features to train an SVM-classifier to identify valid alignments and report an  $F_1$ -score of 0.66 on a manually annotated gold standard. They report 4,265 new English senses and 6,429 new Italian senses, which were derived via MultiWordNet.

*ExtendedWordFramenet* (Laparra and Rigau, 2009; Laparra and Rigau, 2010) is also based on the alignment of FrameNet senses to Word-

Net synsets. The goal is the multilingual coverage extension of FrameNet, which is achieved by linking WordNet to wordnets in other languages (Spanish, Italian, Basque, and Catalan) in the Multilingual Central Repository. For each language, they add more than 10,000 senses to FrameNet. They rely on a knowledge-based word sense disambiguation algorithm to establish the alignment and report  $F_1=0.75$  on a gold standard based on Tonelli and Pighin (2009).

Tonelli and Giuliano (2009) align FrameNet senses to Wikipedia entries with the goal to extract word senses and example sentences in Italian. Based on Wikipedia, this alignment is restricted to nouns. Subsequent work on Wikipedia and FrameNet follows a different path and tries to enhance the modeling of selectional preferences for FrameNet predicates (Tonelli et al., 2012).

Finally, there have been suggestions to combine the corpus-based and the resource-based approaches: Borin et al. (2012) do this for Finnish and Swedish. They prove the feasibility of their approach by creating a preliminary Finnish FrameNet with 2,694 senses.

Mouton et al. (2010) directly exploit the translations in the English and French Wiktionary editions to extend the French FrameNet. They match the FrameNet senses to Wiktionary lexical entries, thus encountering the problem of polysemy in the target language. To solve this, they define a set of filters that control how target lemmas are distributed over frames, increasing precision at the expense of recall ( $P=0.74$ ,  $R=0.3$ ,  $F_1=0.42$ ). While their approach is in theory applicable to other languages, our approach goes beyond this by laying the ground for simultaneous FrameNet extension in multiple languages via FNWKxx.

### 4.2 Wiktionary Sense Alignments

Collaboratively created resources have become popular for sense alignments for NLP, starting with the alignment between WordNet and Wikipedia (Ruiz-Casado et al., 2005; Ponzetto and Navigli, 2009). Wiktionary has been subject to few alignment efforts: de Melo and Weikum (2009) integrate information from Wiktionary into Universal WordNet. Meyer and Gurevych (2011) map WordNet synsets to Wiktionary senses and show their complementary domain coverage.

## 5 FrameNet – Wiktionary Alignment

### 5.1 Alignment Technique

We follow the state-of-the-art sense alignment technique introduced by Niemann and Gurevych (2011). They align senses in WordNet to Wikipedia entries in a supervised setting using semantic similarity measures.

One reason to use their method was that it allows zero alignments or one-to-many alignments. This is crucial for obtaining a high-quality alignment of heterogeneous resources, such as the presented one, because their sense granularity and coverage can diverge a lot.

The alignment algorithm consists of two steps. In the *candidate extraction* step, we iterate over all FrameNet senses and match them with all senses from Wiktionary which have the same lemma and thus are likely to describe the same sense.

This step yields a set of candidate sense pairs  $C_{all}$ . In the *classification* step, a similarity score between the textual information associated with the senses in a candidate pair (e.g., their gloss) is computed and a threshold-based classifier decides for each pair on valid alignments.

Niemann and Gurevych (2011) combine two different types of similarity (i) cosine similarity on bag-of-words vectors (COS) and (ii) a personalized PageRank-based similarity measure (PPR). The PPR measure (Agirre and Soroa, 2009) maps the glosses of the two senses to a semantic vector space spanned up by WordNet synsets and then compares them using the chi-square measure.

The semantic vectors  $\mathbf{ppr}$  are computed using the personalized PageRank algorithm on the WordNet graph. They determine the important nodes in the graph as the nodes that a random walker following the edges visits most frequently:

$$\mathbf{ppr} = cM\mathbf{ppr} + (1 - c)\mathbf{v}_{\mathbf{ppr}}, \quad (1)$$

where  $M$  is a transition probability matrix between the  $n$  WordNet synsets,  $c$  is a damping factor, and  $\mathbf{v}_{\mathbf{ppr}}$  is a vector of size  $n$  representing the probability of jumping to the node  $i$  associated with each  $\mathbf{v}_i$ . For personalized PageRank,  $\mathbf{v}_{\mathbf{ppr}}$  is initialized in a particular way: the initial weight is distributed equally over the  $m$  vector components (i.e., synsets) associated with a word in the sense gloss, other components receive a 0 value.

For each similarity measure, Niemann and Gurevych (2011) determine a threshold ( $t_{ppr}$  and

$t_{cos}$ ) independently on a manually annotated gold standard. The final alignment decision is the conjunction of two decision functions:

$$a(s_s, s_t) = \text{PPR}(s_s, s_t) > t_{ppr} \& \text{COS}(s_s, s_t) > t_{cos}. \quad (2)$$

We differ from Niemann and Gurevych (2011) in that we use a joint training setup which determines  $t_{ppr}$  and  $t_{cos}$  to optimize classification performance directly (as proposed in Gurevych et al. (2012)):

$$(t_{ppr}, t_{cos}) = \text{argmax}_{(t_{ppr}, t_{cos})} F_1(a), \quad (3)$$

where  $F_1$  is the maximized evaluation score and  $a$  is the decision function in equation (2).

### 5.2 Candidate Extraction

To compile the candidate set, we paired senses from both resources with identical lemma-POS combinations. FrameNet senses are defined by a lemma, a gloss, and a frame. Wiktionary senses are defined by a lemma and a gloss. For the FrameNet sense *Activity finish* of the verb *complete*, we find two candidate senses in Wiktionary (*to finish* and *to make whole*). There are on average 3.7 candidates per FrameNet sense. The full candidate set  $C_{all}$  contains over 44,000 sense pairs and covers 97% of the 11,942 FrameNet senses.

### 5.3 Gold Standard Creation

For the gold standard, we sampled 2,900 candidate pairs from  $C_{all}$ . The properties of the gold standard mirror the properties of  $C_{all}$ : the sampling preserved the distribution of POS in  $C_{all}$  (around 40% verbs and nouns, and 12% adjectives) and the average numbers of candidates per FrameNet sense. This ensures that highly polysemous words as well as words with few senses are selected.

Two human raters annotated the sense pairs based on their glosses. The annotation task consisted in a two-class annotation: *Do the presented senses have same meaning - (YES/NO)*. The raters received detailed guidelines and were trained on around 100 sense pairs drawn from the sample.

We computed Cohen's  $\kappa$  to measure the inter-rater agreement between the two raters. It is  $\kappa=0.72$  on the full set, which is considered acceptable according to Artstein and Poesio (2008). An additional expert annotator disambiguated ties.

For comparison: Meyer and Gurevych (2011) report  $\kappa=0.74$  for their WordNet – Wiktionary gold standard, and Niemann and Gurevych (2011)

	adj	noun	verb	all
$\kappa$	<b>.8</b>	<b>.77</b>	<b>.65</b>	<b>.72</b>

Table 1: Inter-rater agreement.

$\kappa=0.87$  for their WordNet – Wikipedia gold standard. These gold standards only consist of nouns, which appear to be an easier annotation task than verb senses. This is supported by our analysis of the agreement by POS (see Table 1): the agreement on nouns and adjectives lies between the two agreement scores previously reported on nouns. Thus our annotation is of similar quality. Only the agreement on verbs is slightly below the acceptability threshold of 0.67 (Artstein and Poesio, 2008). The verb senses are very fine-grained and thus present a difficult alignment task. Therefore, we had an expert annotator correct the verbal part of the gold standard set. After removing the training set for the raters, the final gold standard contains 2,789 sense pairs. 28% of these are aligned.

#### 5.4 Alignment Experiments

We determined the best setting for the alignment of FrameNet and Wiktionary in a ten-fold cross-validation on the gold standard.

Besides the parameters for the computation of the PPR vectors (we used the publicly available UKB tool by Agirre and Soroa (2009)), the main parameter in the experiments is the textual information that is used to represent the senses. For FrameNet senses, we used the *lemma-pos*, *sense gloss*, *example sentences*, *frame name* and *frame definition* as textual features; for Wiktionary senses, we considered *lemma-pos*, *sense gloss*, *example sentences*, *hyponyms* and *synonyms*.

We computed the similarity scores on tokenized, lemmatized and stop-word-filtered texts.

First, we evaluated models for COS and PPR independently based on various combinations of the textual features listed above. We then used the parameter setting of the best-performing single models to train the model that jointly optimizes the thresholds for PPR and COS (see eqn. (5)). In Table 2, we report on the results of the best single models and the best joint model.

For the evaluation, we compute precision P, recall R and  $F_1$  on the positive class (aligned=true), e.g., precision P is the number of pairs correctly aligned divided by all aligned pairs.

We achieved the highest precision and  $F_1$ -score

	Evaluation	verb	noun	adj	all
P	Random-1 BL	0.503	0.559	0.661	0.557
	WKT-1 BL	0.620	0.664	0.725	0.66
	BEST COS	0.639	<b>0.778</b>	0.706	0.703
	BEST PPR	0.66	0.754	0.729	0.713
	BEST JOINT	<b>0.677</b>	0.766	<b>0.742</b>	<b>0.728</b>
R	Random-1 BL	0.471	0.546	0.683	0.540
	WKT-1 BL	0.581	0.65	0.75	0.64
	BEST COS	0.658	0.758	0.754	0.715
	BEST PPR	0.666	0.724	0.754	0.699
	BEST JOINT	<b>0.683</b>	<b>0.783</b>	<b>0.83</b>	<b>0.75</b>
$F_1$	Random-1 BL	0.487	0.552	0.672	0.549
	WKT-1 BL	0.60	0.657	0.737	0.65
	BEST COS	0.648	0.768	0.729	0.709
	BEST PPR	0.663	0.739	0.741	0.706
	BEST JOINT	<b>0.68</b>	<b>0.775</b>	<b>0.784</b>	<b>0.739</b>
	UBound	0.735	0.834	0.864	0.797

Table 2: Alignment performance by POS.

for COS using all available features, but excluding FrameNet *example sentences* because they introduce too much noise. Adding the *frame name* and *frame definition* to the often short glosses provides a richer sense representation for the COS measure.

The best-performing PPR configuration uses *sense gloss* and *lemma-pos*. For the joint model, we employed the best single PPR configuration, and a COS configuration that uses *sense gloss* extended by Wiktionary *hyponyms*, *synonyms* and FrameNet *frame name* and *frame definition*, to achieve the highest score, an  $F_1$ -score of 0.739.

#### 5.5 Gold Standard Evaluation

We compared the performance of our alignment on the gold standard to a baseline which randomly selects one target sense from the candidate set of each source sense (Random-1). We also consider the more competitive Wiktionary first sense baseline (WKT-1). It is guided by the heuristic that more frequent senses are listed first in Wiktionary (Meyer and Gurevych, 2010). It is a stronger baseline with an  $F_1$ -score of 0.65 (see Table 2).

To derive the upper bound for the alignment performance (UBound), we computed the  $F_1$  score from the average pairwise  $F_1$ -score of the annotators according to Hripcsak and Rothschild (2005).

As the evaluation set mirrors the POS distribution in FrameNet and is sufficiently large, unlike earlier alignments, we can analyze the performance by POS. The BEST JOINT model performs well on nouns, slightly better on adjectives, and worse on verbs, see Table 2. For the baselines and the UBound the same applies, with the difference that adjectives receive even better results

in comparison. This fits in with the perceived degree of difficulty according to the observed polysemy for the POS: for verbs we have many candidate sets with two or more candidates, i.e., we observe higher polysemy, while for nouns and even stronger for adjectives, many small candidate sets occur, which stand for an easier alignment decision. This is in line with the reported higher complexity of lexical resources with respect to verbs and greater difficulty in alignments and word sense disambiguation (Laparra and Rigau, 2010).

The performance of BEST JOINT on all POS is  $F_1=0.73$ , which is significantly higher than the WKT-1 baseline ( $p<0.05$  according to McNemar’s test). The performance on nouns ( $F_1=0.775$ ) is on par with the results reported by Niemann and Gurevych (2011) for nouns ( $F_1=0.78$ ).

## 5.6 Error Analysis

The confusion matrix from the evaluation of BEST JOINT on the gold standard shows 214 false positives and 191 false negatives. The false negatives suffer from low overlap between the glosses, which are often quite short (*contend - assert*), sometimes circular (*sinful - relating to sin*). Aligning senses with such glosses is difficult for a system based on semantic similarity. In about 50% of the analyzed pairs, highly similar words are used in the gloss, that we should be able to detect with second-order representations, for instance by expanding short definitions with the definitions of the contained words, or via derivational similarity.

A number of false positives occur because the gold standard was developed in a very fine-grained manner: distinctions such as causative vs. inchoative (*enlarge: become large vs. enlarge: make large*) were explicitly stressed in the definitions and thus annotated as different senses by the annotators. This was motivated by the fact that this distinction is relevant for many frames in FrameNet. The first meaning of *enlarge* belongs to the frame *Expansion*, the second to *Cause expansion*. Our similarity based approach cannot capture such differences well.

## 6 Intermediate Resource FNWKxx

### 6.1 Statistics

We applied the best system setup to the full candidate set of over 44,000 candidates to create the intermediate resource FNWKxx. The alignment consists of 12,094 sense pairs. It covers 82% of

	fine-grained P			coarse-grained P		
All POS	0.67			0.78		
By POS	verb	noun	adj	verb	noun	adj
	0.53	0.73	0.80	0.73	0.82	0.85

Table 3: Post-hoc evaluation (precision P).

the senses in FrameNet and 86% of the frames. It connects more than 9,800 unique FrameNet senses with more than 10,000 unique Wiktionary senses, which shows that both non-alignments and multiple alignments occur for some source senses.

## 6.2 Post-hoc Evaluation

Our cross-validation approach entails the danger of over-fitting. In order to verify the quality of the alignment, we performed a detailed post-hoc analysis on a sample of 270 aligned sense pairs randomly drawn from the set of aligned senses.

Because sense granularity was an issue in the error analysis, we considered two alignment decisions: (a) fine-grained alignment: the two glosses describe the same sense; (b) coarse-grained alignment. The causative/inchoative distinction is, among others, ignored.

The evaluation results are listed in Table 3. The precision for the fine-grained (a) is lower than the all-over precision on the gold standard. The evaluation by POS shows that the result for nouns and adjectives is equal or superior to the evaluation result on the gold standard, while it is worse for verbs. This shows that over-fitting, if at all, is only a risk for the verb senses.

The all-over precision for (b) exceeds the precision on the gold standard. Particularly verbs receive much better results. This shows that a coarse-grained alignment may suffice for the FrameNet extension.

This evaluation confirms the quality of the sense alignment, in particular with respect to the FrameNet extension. But it also elicits the question whether a coarse-grained alignment would suffice. We will discuss this question below.

## 6.3 Resource Analysis

For each of the aligned senses in the 12,094 aligned sense pairs, we extracted **glosses** from Wiktionary. Because FrameNet glosses are often very brief, the additional glosses will benefit algorithms such as frame detection for SRL. We also added 4,352 new **example sentences** from Wik-

tionary to FrameNet.

We can extract 2,151 new **lemma-POS** for FrameNet frames from the synonyms of the aligned senses in Wiktionary. We also extract other related lemma-POS, for instance 487 antonyms, 126 hyponyms, and 19 hypernyms.

This step establishes Wiktionary as an interlingual connection between FrameNet and a large number of languages, including low-resource ones: via Wiktionary, we connect FrameNet senses to **translations** in 283 languages, e.g., we translate the sense of the verb *complete* associated with the frame *Activity Finish* to the German colloquial *fertigmachen*, the Spanish *terminar*, the Turkish *tamamlamak*, and 19 other languages.

For 36 languages, we can extract more than 1,000 translations each, among them low-resource languages such as Telugu, Swahili, or Kurdish. The languages with most translations are: Finnish (9,333), Russian (7,790), and German (6,871). The number of Finnish translations is more than three times larger than the preliminary Finnish FrameNet by Borin et al. (2012). Likewise, we get three times the number of German lemma-POS than provided by the SALSA corpus.

## 7 Translation Disambiguation

### 7.1 Disambiguation Method

FNWKxx initially does not provide lexical-semantic information for the German translations: the translations link to a lemma in the German Wiktionary, not a target sense. In order to integrate the information attached to a German Wiktionary sense, e.g., the gloss, into our resource, the lemmas need to be disambiguated.

We use the sense-disambiguated Wiktionary resulting from a recently published approach for the disambiguation of relations and translations in Wiktionary (Meyer and Gurevych, 2012a) to create our new bilingual (German-English) FrameNet lexicon FNWKde.

Their approach combines information on the source sense and all potential target senses in order to determine the best target sense in a rule-based disambiguation strategy. The information is encoded as binary features, which are ordered in a back-off hierarchy: if the first feature applies, the target sense is selected, otherwise the second feature is considered, and so forth.

The most important features are: definition overlap between source and automatically trans-

	SALSA2	P&L05	FNWKde
Type	Corpus	Corpus	Lexicon
Creation	Manual	Automatic	Automatic
Frames(+p)	266(907)	468	755
Senses	1,813	9,851	5,897
Examples	24,184	1,672,551	6,933
Glosses	-	-	5,897

Table 4: Frame-semantic resources for German.

lated target definition; occurrence of the source lemma in the target definition; shared linguistic information (e.g., same register); inverse translation relations (i.e., the source lemma occurs on the translation list of the target sense); relation overlap; Lesk measure between original and translated glosses in source and target language; and finally, backing off to the first target sense.

For the gold standard evaluation of the approach we refer to Meyer and Gurevych (2012a): their system obtained an  $F_1$ -score of 0.67 for the task of disambiguating translations from English to German, and an  $F_1$ -score of 0.79 for the disambiguation of English sense relations. We use the latter to identify target senses of synonyms in FNWKxx.

## 8 Resource FNWKde

### 8.1 Statistics

Table 4 gives an overview of FNWKde. It contains 5,897 pairs of German Wiktionary senses and FrameNet senses, i.e., 86% of the translations could be disambiguated. Each sense has a gloss, and there are 6,933 example sentences.

Based on the relation disambiguation and inference of new relations by Meyer and Gurevych (2012a), we can also disambiguate synonyms in the English Wiktionary. This leads to a further extension of the English FrameNet summarized in Table 5. The number of Wiktionary senses aligned to FrameNet senses is increased by 50%.

We also provide results for other sense relations, e.g., antonyms. We will discuss whether and how they can be integrated as FrameNet senses in our resource below.

### 8.2 Post-hoc Evaluation

Because the errors of two subsequently applied automatic methods can multiply, we provide a post-hoc evaluation of the results.

To evaluate the quality of the German FrameNet lexicon, we collected the FrameNet senses for a list of 15 frames that were sampled by Padó and

Relation	# English senses per FrameNet sense	# English senses per frame
SYNONYM	17,713	13,288
HYPONYM	4,818	3,347
HYPERNYM	6,369	3,961
ANTONYM	9,626	6,737

Table 5: Statistics after relation disambiguation.

Lapata (2005) according to three frequency bands on a large corpus. There are 115 senses associated with these frames in our resource. In a manual evaluation of these 115 senses, we find that 67% were assigned correctly to their frames. This is higher than expected, considering the errors from the applied methods add up.

Further analysis revealed that both resource creation steps contribute equally to the 39 errors. For 17 of the evaluated sense pairs, redundancy confirms their quality: they were obtained independently by two or three alignment-and-translation paths and do not contain alignment errors.

### 8.3 Comparison

We compare FNWKde to two German frame-semantic resources, the manually annotated SALSA corpus (Burchardt et al., 2006) and a resource from Padó and Lapata (2005), henceforth P&L05. Note that both resources are frame-annotated corpora, while FNWKde is a FrameNet-like *lexicon* and contains information complementary to the corpora. The different properties of the resources are contrasted in Table 4.

The automatically developed resources, including FNWKde, provide a larger number of senses than SALSA. The annotated corpora contain a large number of examples, but they do not provide any glosses, which are useful for frame detection in SRL, nor do they contain any other lexical-semantic information.

FNWKde covers a larger number of FrameNet frames than the other two resources. 266 of the 907 frames in SALSA are connected to original FrameNet frames, the others are newly-developed proto-frames *p* (shown in parentheses in Table 4).

Table 6 describes the proportion of the overlapping frames and senses<sup>3</sup> to the respective resources. The numbers on frame overlap show that our resource covers the frames in the other

<sup>3</sup>Note that the senses in SALSA and P&L05 are defined by frame, lemma, and POS. In Table 6, FNWKde senses with identical frame, lemma, and POS, but different gloss are therefore conflated to one sense.

	Resource r	% of r	% of FNWKde
Frame	SALSA 2	89%	31%
	P&L05	90%	55%
Sense	SALSA 2	15%	5%
	P&L05	10%	19%

Table 6: Overlap of FNWKde with resource r.

resources well (89% and 90% coverage respectively), and that it adds frames not covered in the other resources: P&L05 only covers 55% of the frames in FNWKde. The sense overlap shows that the resources have senses in common, which confirms the quality of the automatically developed resources, but they also complement each other. FNWKde, for instance, adds 3,041 senses to P&L05.

## 9 Discussion: a Multilingual FrameNet

FNWKxx builds an excellent starting point to create FrameNet lexicons in various languages: the translation counts, for instance 6,871 for German, compare favorably to FrameNet 1.5, which contains 9,700 English lemma-POS.

To create those FrameNet lexicons, the translation disambiguation approach used for FNWKde (step 2 in Fig. 1) needs to be adapted to other languages. The approach is in theory applicable to any language, but there are some obstacles: first, it relies on the availability of the target sense in the target language Wiktionary. For many of the top 30 languages in FNWKxx, the Wiktionary editions seem sufficiently large to provide targets for translation disambiguation,<sup>4</sup> and they are continuously extended. Second, our approach requires access to the target language Wiktionary, but the data format across Wiktionary language editions is not standardized. Third, the approach requires machine translation into the target language. For languages, where such a tool is not available, we could default to the first-sense-heuristic, or encourage the Wiktionary community to link the translations to their target Wiktionary senses inspired by Sajous et al. (2010).

Another issue that applies to all automatic (and also manual) approaches of cross-lingual FrameNet extension is the restricted cross-language applicability of frames. Boas (2005) reports that, while many frames are largely

<sup>4</sup>see overview table at <http://www.ukp.tu-darmstadt.de/fnwkde/>.

language-independent, other frames receive culture-specific or language-specific interpretations, for example calendars or holidays. Also, fine-grained sense and frame distinctions may be more relevant in one language than in another language. Such granularity differences also led to the addition of proto-frames in SALSA 2 (Rehbein et al., 2012). Therefore, manual correction or extension of a multilingual FrameNet based on FNWKde may be desired for specific applications. In this case, the automatically created FrameNets in other languages are good starting points that can be quickly and efficiently compiled.

The quality of the multilingual FNWKxx depends on i) the translations in the interlingual connection Wiktionary, which are manually created, controlled by the community, and therefore reliable, and ii) on the FrameNet–Wiktionary alignment. Therefore, we evaluated our sense alignment method in detail. The alignment reaches state-of-the-art results, and the analysis shows that the method is particularly fit for a coarse-grained alignment. We however find lower performance for verbs in a fine-grained setting. We argue that an improved alignment algorithm, for instance taking subcategorization information into account, can identify the fine-grained distinctions.

The post-hoc analysis raised the question of FrameNet frame granularity. Do separate frames exist for causative/inchoative alternations (as *Being dry* and *Cause to be dry* for *to dry*), or do they belong to the same frame (*Make noise* for *to creak* and *to creak something*)? For the coarse-grained frames, fine-grained decisions can be merged in a second classification step. Alternatively, we could map Wiktionary senses directly to frames, and include features that cover the granularity distinctions, e.g., whether the existing senses of a frame show the semantic alternation.

We could use the same approach to assign senses to a frame which are derived via sense relations other than synonymy, i.e., for linking antonyms or hyponyms to a frame. Some frames do cover antonymous predicates, others do not.

Based on Wiktionary, our approach suffers less from the disadvantages of previous resource-based work, i.e., the constraints of expert-built resources and the lack of lexical information in Wikipedia. Unlike corpus-based approaches for cross-lingual FrameNet extension, our approach does not provide frame-semantic annotations for the example

sentences. Our advantage is that we create a FrameNet *lexicon* with lexical-semantic information in the target language. Example annotations can be additionally obtained via cross-lingual annotation projection (Padó and Lapata, 2009), and the lexical information in FNWKde can be used to guide this process.

## 10 Conclusion

The resource-coverage bottleneck for frame-semantic resources is particularly severe for less well-resourced languages. We present a simple, but effective approach to solve this problem using the English Wiktionary as an interlingual representation and subsequent translation disambiguation in the target language. We validate our approach on the language pair English-German and discuss the options and requirements for creating FrameNets in further languages.

As part of this work, we created the first sense alignment between FrameNet and the English Wiktionary. The resulting resource FNWKxx connects FrameNet senses to over 280 languages. The bilingual English-German FrameNet lexicon FNWKde competes with manually created resources, as shown by a comparison to the SALSA corpus.

We make both resources publicly available in the standardized format UBY-LMF (Eckle-Kohler et al., 2012), which supports automatic processing of the resources via the UBY Java API, see <http://www.ukp.tu-darmstadt.de/fnwkde/>.

We also extended FrameNet by several thousand new English senses from Wiktionary which are provided as part of FNWKde. In our future work, we will evaluate the benefits of the extracted information to SRL.

## Acknowledgments

This work has been supported by the Volkswagen Foundation as part of the Lichtenberg-Professorship Program under grant No. I/82806 and by the German Research Foundation under grant No. GU 798/3-1 and grant No. GU 798/9-1.

We thank Christian Meyer and Judith-Eckle Kohler for insightful discussions and comments, and Christian Wirth for contributions in the early stage of this project. We also thank the anonymous reviewers for their helpful remarks.

## References

- Eneko Agirre and Aitor Soroa. 2009. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41, Athens, Greece.
- Ron Artstein and Massimo Poesio. 2008. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596.
- Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL'98)*, pages 86–90, Montreal, Canada.
- Roberto Basili, Diego Cao, Danilo Croce, Bonaventura Coppola, and Alessandro Moschitti. 2009. Cross-language frame semantics transfer in bilingual corpora. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 5449 of *Lecture Notes in Computer Science*, pages 332–345. Springer Berlin Heidelberg.
- Hans C. Boas. 2005. Semantic Frames as Interlingual Representations for Multilingual Lexical Databases. *International Journal of Lexicography*, 18(4):445–478.
- Lars Borin, Markus Forsberg, Richard Johansson, Kristiina Muhonen, Tanja Purtonen, and Kaarlo Voionmaa. 2012. Transferring frames: Utilization of linked lexical resources. In *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 8–15, Montréal, Canada.
- Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Pado, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 969–974, Genoa, Italy.
- Diego De Cao, Danilo Croce, Marco Pennacchiotti, and Roberto Basili. 2008. Combining word sense and usage for modeling frame semantics. In *Proceedings of the 2008 Conference on Semantics in Text Processing*, STEP '08, pages 85–101, Stroudsburg, PA, USA.
- Gerard de Melo and Gerhard Weikum. 2009. Towards a universal wordnet by learning from combined evidence. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 513–522, New York, NY, USA.
- Judith Eckle-Kohler, Iryna Gurevych, Silvana Hartmann, Michael Matuschek, and Christian M. Meyer. 2012. UBY-LMF - A Uniform Model for Standardizing Heterogeneous Lexical-Semantic Resources in ISO-LMF. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12)*, pages 275–282, Istanbul, Turkey.
- Oscar Ferrandez, Michael Ellsworth, Rafael Munoz, and Collin F. Baker. 2010. Aligning FrameNet and WordNet based on Semantic Neighborhoods. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 310–314, Valletta, Malta.
- Charles J. Fillmore. 1976. Frame Semantics and the Nature of Language. In *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language and Speech*, volume 280, pages 20–32. New York Academy of Sciences, New York, NY, USA.
- Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. 2012. Uby - A Large-Scale Unified Lexical-Semantic Resource Based on LMF. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 580–590, Avignon, France.
- George Hripcsak and Adam S. Rothschild. 2005. Agreement, the F-Measure, and Reliability in Information Retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298.
- Richard Johansson and Pierre Nugues. 2005. Using Parallel Corpora for Automatic Transfer of FrameNet Annotation. In *Proceedings of the 1st ROMANCE FrameNet Workshop*, Cluj-Napoca, Romania.
- Richard Johansson and Pierre Nugues. 2006. A framenet-based semantic role labeler for swedish. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 436–443, Sydney, Australia, July.
- Richard Johansson and Pierre Nugues. 2007. Using WordNet to extend FrameNet coverage. In *Proceedings of the Workshop on Building Frame-semantic Resources for Scandinavian and Baltic Languages, at NODALIDA*, pages 27–30, Tartu, Estonia.
- Egoitz Laparra and German Rigau. 2009. Integrating WordNet and FrameNet using a Knowledge-based Word Sense Disambiguation Algorithm. In *Proceedings of the International Conference RANLP-2009*, pages 208–213, Borovets, Bulgaria.
- Egoitz Laparra and German Rigau. 2010. eXtended WordFrameNet. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 1214–1419, Valletta, Malta.
- Christian M. Meyer and Iryna Gurevych. 2010. How Web Communities Analyze Human Language: Word Senses in Wiktionary. In *Proceedings of the Second Web Science Conference*, Raleigh, NC, USA.

- Christian M. Meyer and Iryna Gurevych. 2011. What Psycholinguists Know About Chemistry: Aligning Wiktionary and WordNet for Increased Domain Coverage. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 883–892, Chiang Mai, Thailand.
- Christian M. Meyer and Iryna Gurevych. 2012a. To Exhibit is not to Loiter: A Multilingual, Sense-Disambiguated Wiktionary for Measuring Verb Similarity. In *Proceedings of COLING 2012*, pages 1763–1780, Mumbai, India.
- Christian M. Meyer and Iryna Gurevych. 2012b. Wiktionary: A new rival for expert-built lexicons? Exploring the possibilities of collaborative lexicography. In Sylviane Granger and Magali Paquot, editors, *Electronic Lexicography*, pages 259–291. Oxford University Press, Oxford.
- Behrang Mohit and Sridhar Narayanan. 2003. Semantic Extraction with Wide-Coverage Lexical Resources. In *Proceedings of HLT-NAACL 2003: Companion Volume*, pages 64–66, Edmonton, Canada.
- Claire Mouton, Gaël de Chalendar, and Benoît Richert. 2010. FrameNet Translation Using Bilingual Dictionaries with Evaluation on the English-French Pair. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pages 20–27, Valletta, Malta.
- Sridhar Narayanan and Sanda Harabagiu. 2004. Question Answering Based on Semantic Structures. In *Proceedings of the 20th international conference on Computational Linguistics - COLING '04*, pages 693–701, Geneva, Switzerland.
- Roberto Navigli and Simone Paolo Ponzetto. 2010. Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 216–225, Uppsala, Sweden.
- Elisabeth (geb. Wolf) Niemann and Iryna Gurevych. 2011. The People’s Web meets Linguistic Knowledge: Automatic Sense Alignment of Wikipedia and WordNet. In *Proceedings of the International Conference on Computational Semantics (IWCS)*, pages 205–214, Singapore.
- Sebastian Padó and Mirella Lapata. 2005. Cross-lingual bootstrapping of semantic lexicons: the case of FrameNet. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 3, AAAI'05*, pages 1087–1092, Pittsburgh, PA, USA.
- Sebastian Padó and Mirella Lapata. 2009. Cross-lingual Annotation Projection for Semantic Roles. *Journal of Artificial Intelligence Research*, 36:307–340.
- Simone Paolo Ponzetto and Roberto Navigli. 2009. Large-Scale Taxonomy Mapping for Restructuring and Integrating Wikipedia. In *Proceedings of the 21st International Joint Conference on AI*, pages 2083–2088, Pasadena, CA, USA.
- Ines Rehbein, Joseph Ruppenhofer, Caroline Sporleder, and Manfred Pinkal. 2012. Adding nominal spice to SALSA - frame-semantic annotation of German nouns and verbs. In *Proceedings of the 11th Conference on Natural Language Processing (KONVENS'12)*, pages 89–97, Vienna, Austria.
- Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. 2005. Automatic Assignment of Wikipedia Encyclopedic Entries to WordNet Synsets. In *Advances in Web Intelligence*, volume 3528 of *Lecture Notes in Computer Science*, pages 380–386. Springer, Berlin Heidelberg.
- Franck Sajous, Emmanuel Navarro, Bruno Gaume, Laurent Prévot, and Yannick Chudy. 2010. Semi-automatic endogenous enrichment of collaboratively constructed lexical resources: piggybacking onto wiktionary. In *Proceedings of the 7th international conference on Advances in natural language processing, IceTAL'10*, pages 332–344. Springer, Berlin, Heidelberg.
- Lei Shi and Rada Mihalcea. 2005. Putting pieces together: Combining FrameNet, VerbNet and WordNet for robust semantic parsing. In *Computational Linguistics and Intelligent Text Processing*, pages 100–111. Springer, Berlin Heidelberg.
- Sara Tonelli and Claudio Giuliano. 2009. Wikipedia as frame information repository. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 276–285, Singapore.
- Sara Tonelli and Emanuele Pianta. 2009. A novel approach to mapping FrameNet lexical units to WordNet synsets. In *IWCS-8 '09: Proceedings of the Eighth International Conference on Computational Semantics*, pages 342–345, Tilburg, The Netherlands.
- Sara Tonelli and Daniele Pighin. 2009. New Features for FrameNet - WordNet Mapping. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 219–227, Boulder, CO, USA.
- Sara Tonelli, Volha Bryl, Claudio Giuliano, and Luciano Serafini. 2012. Investigating the semantics of frame elements. In *Knowledge Engineering and Knowledge Management*, volume 7603 of *Lecture Notes in Computer Science*, pages 130–143. Springer Berlin Heidelberg.

# Dirt Cheap Web-Scale Parallel Text from the Common Crawl

**Jason R. Smith**<sup>1,2</sup>  
jsmith@cs.jhu.edu

**Herve Saint-Amand**<sup>3</sup>  
herve@saintamh.org

**Magdalena Plamada**<sup>4</sup>  
plamada@cl.uzh.ch

**Philipp Koehn**<sup>3</sup>  
pkoehn@inf.ed.ac.uk

**Chris Callison-Burch**<sup>1,2,5</sup>  
ccb@cs.jhu.edu \*

**Adam Lopez**<sup>1,2</sup>  
alopez@cs.jhu.edu

<sup>1</sup>Department of Computer Science, Johns Hopkins University

<sup>2</sup>Human Language Technology Center of Excellence, Johns Hopkins University

<sup>3</sup>School of Informatics, University of Edinburgh

<sup>4</sup>Institute of Computational Linguistics, University of Zurich

<sup>5</sup>Computer and Information Science Department, University of Pennsylvania

## Abstract

Parallel text is the fuel that drives modern machine translation systems. The Web is a comprehensive source of preexisting parallel text, but crawling the entire web is impossible for all but the largest companies. We bring web-scale parallel text to the masses by mining the Common Crawl, a public Web crawl hosted on Amazon's Elastic Cloud. Starting from nothing more than a set of common two-letter language codes, our open-source extension of the STRAND algorithm mined 32 terabytes of the crawl in just under a day, at a cost of about \$500. Our large-scale experiment uncovers large amounts of parallel text in dozens of language pairs across a variety of domains and genres, some previously unavailable in curated datasets. Even with minimal cleaning and filtering, the resulting data boosts translation performance across the board for five different language pairs in the news domain, and on open domain test sets we see improvements of up to 5 BLEU. We make our code and data available for other researchers seeking to mine this rich new data resource.<sup>1</sup>

## 1 Introduction

A key bottleneck in porting statistical machine translation (SMT) technology to new languages and domains is the lack of readily available parallel corpora beyond curated datasets. For a handful of language pairs, large amounts of parallel data

are readily available, ordering in the hundreds of millions of words for Chinese-English and Arabic-English, and in tens of millions of words for many European languages (Koehn, 2005). In each case, much of this data consists of government and news text. However, for most language pairs and domains there is little to no curated parallel data available. Hence discovery of parallel data is an important first step for translation between most of the world's languages.

The Web is an important source of parallel text. Many websites are available in multiple languages, and unlike other potential sources—such as multilingual news feeds (Munteanu and Marcu, 2005) or Wikipedia (Smith et al., 2010)—it is common to find document pairs that are direct translations of one another. This natural parallelism simplifies the mining task, since few resources or existing corpora are needed at the outset to bootstrap the extraction process.

Parallel text mining from the Web was originally explored by individuals or small groups of academic researchers using search engines (Nie et al., 1999; Chen and Nie, 2000; Resnik, 1999; Resnik and Smith, 2003). However, anything more sophisticated generally requires direct access to web-crawled documents themselves along with the computing power to process them. For most researchers, this is prohibitively expensive. As a consequence, web-mined parallel text has become the exclusive purview of large companies with the computational resources to crawl, store, and process the entire Web.

To put web-mined parallel text back in the hands of individual researchers, we mine parallel text from the Common Crawl, a regularly updated 81-terabyte snapshot of the public internet hosted

\*This research was conducted while Chris Callison-Burch was at Johns Hopkins University.

<sup>1</sup>[github.com/jrs026/CommonCrawlMiner](https://github.com/jrs026/CommonCrawlMiner)

on Amazon’s Elastic Cloud (EC2) service.<sup>2</sup> Using the Common Crawl completely removes the bottleneck of web crawling, and makes it possible to run algorithms on a substantial portion of the web at very low cost. Starting from nothing other than a set of language codes, our extension of the STRAND algorithm (Resnik and Smith, 2003) identifies potentially parallel documents using cues from URLs and document content (§2). We conduct an extensive empirical exploration of the web-mined data, demonstrating coverage in a wide variety of languages and domains (§3). Even without extensive pre-processing, the data improves translation performance on strong baseline news translation systems in five different language pairs (§4). On general domain and speech translation tasks where test conditions substantially differ from standard government and news training text, web-mined training data improves performance substantially, resulting in improvements of up to 1.5 BLEU on standard test sets, and 5 BLEU on test sets outside of the news domain.

## 2 Mining the Common Crawl

The Common Crawl corpus is hosted on Amazon’s Simple Storage Service (S3). It can be downloaded to a local cluster, but the transfer cost is prohibitive at roughly 10 cents per gigabyte, making the total over \$8000 for the full dataset.<sup>3</sup> However, it is unnecessary to obtain a copy of the data since it can be accessed freely from Amazon’s Elastic Compute Cloud (EC2) or Elastic MapReduce (EMR) services. In our pipeline, we perform the first step of identifying candidate document pairs using Amazon EMR, download the resulting document pairs, and perform the remaining steps on our local cluster. We chose EMR because our candidate matching strategy fit naturally into the Map-Reduce framework (Dean and Ghemawat, 2004).

Our system is based on the STRAND algorithm (Resnik and Smith, 2003):

1. *Candidate pair selection*: Retrieve candidate document pairs from the CommonCrawl corpus.
2. *Structural Filtering*:
  - (a) Convert the HTML of each document

into a sequence of start tags, end tags, and text chunks.

- (b) Align the linearized HTML of candidate document pairs.
  - (c) Decide whether to accept or reject each pair based on features of the alignment.
3. *Segmentation*: For each text chunk, perform sentence and word segmentation.
  4. *Sentence Alignment*: For each aligned pair of text chunks, perform the sentence alignment method of Gale and Church (1993).
  5. *Sentence Filtering*: Remove sentences that appear to be boilerplate.

**Candidate Pair Selection** We adopt a strategy similar to that of Resnik and Smith (2003) for finding candidate parallel documents, adapted to the parallel architecture of Map-Reduce.

The *mapper* operates on each website entry in the CommonCrawl data. It scans the URL string for some indicator of its language. Specifically, we check for:

1. Two/three letter language codes (ISO-639).
2. Language names in English and in the language of origin.

If either is present in a URL and surrounded by non-alphanumeric characters, the URL is identified as a potential match and the mapper outputs a key value pair in which the key is the original URL with the matching string replaced by \*, and the value is the original URL, language name, and full HTML of the page. For example, if we encounter the URL `www.website.com/fr/`, we output the following.

- Key: `www.website.com/*/`
- Value: `www.website.com/fr/`, French, (full website entry)

The *reducer* then receives all websites mapped to the same “language independent” URL. If two or more websites are associated with the same key, the reducer will output all associated values, as long as they are not in the same language, as determined by the language identifier in the URL.

This URL-based matching is a simple and inexpensive solution to the problem of finding candidate document pairs. The mapper will discard

<sup>2</sup>[commoncrawl.org](http://commoncrawl.org)

<sup>3</sup><http://aws.amazon.com/s3/pricing/>

most, and neither the mapper nor the reducer do anything with the HTML of the documents aside from reading and writing them. This approach is very simple and likely misses many good potential candidates, but has the advantage that it requires no information other than a set of language codes, and runs in time roughly linear in the size of the dataset.

**Structural Filtering** A major component of the STRAND system is the alignment of HTML documents. This alignment is used to determine which document pairs are actually parallel, and if they are, to align pairs of text blocks within the documents.

The first step of structural filtering is to linearize the HTML. This means converting its DOM tree into a sequence of start tags, end tags, and chunks of text. Some tags (those usually found within text, such as “font” and “a”) are ignored during this step. Next, the tag/chunk sequences are aligned using dynamic programming. The objective of the alignment is to maximize the number of matching items.

Given this alignment, Resnik and Smith (2003) define a small set of features which indicate the alignment quality. They annotated a set of document pairs as parallel or non-parallel, and trained a classifier on this data. We also annotated 101 Spanish-English document pairs in this way and trained a maximum entropy classifier. However, even when using the best performing subset of features, the classifier only performed as well as a naive classifier which labeled every document pair as parallel, in both accuracy and F1. For this reason, we excluded the classifier from our pipeline. The strong performance of the naive baseline was likely due to the unbalanced nature of the annotated data— 80% of the document pairs that we annotated were parallel.

**Segmentation** The text chunks from the previous step may contain several sentences, so before the sentence alignment step we must perform sentence segmentation. We use the Punkt sentence splitter from NLTK (Loper and Bird, 2002) to perform both sentence and word segmentation on each text chunk.

**Sentence Alignment** For each aligned text chunk pair, we perform sentence alignment using the algorithm of Gale and Church (1993).

**Sentence Filtering** Since we do not perform any boilerplate removal in earlier steps, there are many sentence pairs produced by the pipeline which contain menu items or other bits of text which are not useful to an SMT system. We avoid performing any complex boilerplate removal and only remove segment pairs where either the source and target text are identical, or where the source or target segments appear more than once in the extracted corpus.

### 3 Analysis of the Common Crawl Data

We ran our algorithm on the 2009-2010 version of the crawl, consisting of 32.3 terabytes of data. Since the full dataset is hosted on EC2, the only cost to us is CPU time charged by Amazon, which came to a total of about \$400, and data storage/transfer costs for our output, which came to roughly \$100. For practical reasons we split the run into seven subsets, on which the full algorithm was run independently. This is different from running a single Map-Reduce job over the entire dataset, since websites in different subsets of the data cannot be matched. However, since the data is stored as it is crawled, it is likely that matching websites will be found in the same split of the data. Table 1 shows the amount of raw parallel data obtained for a large selection of language pairs.

As far as we know, ours is the first system built to mine parallel text from the Common Crawl. Since the resource is new, we wanted to understand the quantity, quality, and type of data that we are likely to obtain from it. To this end, we conducted a number of experiments to measure these features. Since our mining heuristics are very simple, these results can be construed as a lower bound on what is actually possible.

#### 3.1 Recall Estimates

Our first question is about recall: of all the possible parallel text that is actually available on the Web, how much does our algorithm actually find in the Common Crawl? Although this question is difficult to answer precisely, we can estimate an answer by comparing our mined URLs against a large collection of previously mined URLs that were found using targeted techniques: those in the French-English Gigaword corpus (Callison-Burch et al., 2011).

We found that 45% of the URL pairs would

	<b>French</b>	<b>German</b>	<b>Spanish</b>	<b>Russian</b>	<b>Japanese</b>	<b>Chinese</b>
Segments	10.2M	7.50M	5.67M	3.58M	1.70M	1.42M
Source Tokens	128M	79.9M	71.5M	34.7M	9.91M	8.14M
Target Tokens	118M	87.5M	67.6M	36.7M	19.1M	14.8M
	<b>Arabic</b>	<b>Bulgarian</b>	<b>Czech</b>	<b>Korean</b>	<b>Tamil</b>	<b>Urdu</b>
Segments	1.21M	909K	848K	756K	116K	52.1K
Source Tokens	13.1M	8.48M	7.42M	6.56M	1.01M	734K
Target Tokens	13.5M	8.61M	8.20M	7.58M	996K	685K
	<b>Bengali</b>	<b>Farsi</b>	<b>Telugu</b>	<b>Somali</b>	<b>Kannada</b>	<b>Pashto</b>
Segments	59.9K	44.2K	50.6K	52.6K	34.5K	28.0K
Source Tokens	573K	477K	336K	318K	305K	208K
Target Tokens	537K	459K	358K	325K	297K	218K

Table 1: The amount of parallel data mined from CommonCrawl for each language paired with English. Source tokens are counts of the foreign language tokens, and target tokens are counts of the English language tokens.

have been discovered by our heuristics, though we actually only find 3.6% of these URLs in our output.<sup>4</sup> If we had included “f” and “e” as identifiers for French and English respectively, coverage of the URL pairs would increase to 74%. However, we chose not to include single letter identifiers in our experiments due to the high number of false positives they generated in preliminary experiments.

### 3.2 Precision Estimates

Since our algorithms rely on cues that are mostly external to the contents of the extracted data and have no knowledge of actual languages, we wanted to evaluate the precision of our algorithm: how much of the mined data actually consists of parallel sentences?

To measure this, we conducted a manual analysis of 200 randomly selected sentence pairs for each of three language pairs. The texts are heterogeneous, covering several topical domains like tourism, advertising, technical specifications, finances, e-commerce and medicine. For German-English, 78% of the extracted data represent perfect translations, 4% are paraphrases of each other (convey a similar meaning, but cannot be used for SMT training) and 18% represent misalignments. Furthermore, 22% of the true positives are potentially machine translations (judging by the quality), whereas in 13% of the cases one of the sentences contains additional content not ex-

<sup>4</sup>The difference is likely due to the coverage of the CommonCrawl corpus.

pressed in the other. As for the false positives, 13.5% of them have either the source or target sentence in the wrong language, and the remaining ones representing failures in the alignment process. Across three languages, our inspection revealed that around 80% of randomly sampled data appeared to contain good translations (Table 2). Although this analysis suggests that language identification and SMT output detection (Venugopal et al., 2011) may be useful additions to the pipeline, we regard this as reasonably high precision for our simple algorithm.

Language	Precision
Spanish	82%
French	81%
German	78%

Table 2: Manual evaluation of precision (by sentence pair) on the extracted parallel data for Spanish, French, and German (paired with English).

In addition to the manual evaluation of precision, we applied language identification to our extracted parallel data for several additional languages. We used the “langid.py” tool (Lui and Baldwin, 2012) at the segment level, and report the percentage of sentence pairs where both sentences were recognized as the correct language. Table 3 shows our results. Comparing against our manual evaluation from Table 2, it appears that many sentence pairs are being incorrectly judged as non-parallel. This is likely because language identification tends to perform poorly on short segments.

French	German	Spanish	Arabic
63%	61%	58%	51%
Chinese	Japanese	Korean	Czech
50%	48%	48%	47%
Russian	Urdu	Bengali	Tamil
44%	31%	14%	12%
Kannada	Telugu	Kurdish	
12%	6.3%	2.9%	

Table 3: Automatic evaluation of precision through language identification for several languages paired with English.

### 3.3 Domain Name and Topic Analysis

Although the above measures tell us something about how well our algorithms perform in aggregate for specific language pairs, we also wondered about the actual contents of the data. A major difficulty in applying SMT even on languages for which we have significant quantities of parallel text is that most of that parallel text is in the news and government domains. When applied to other genres, such systems are notoriously brittle. What kind of genres are represented in the Common Crawl data?

We first looked at the domain names which contributed the most data. Table 4 gives the top five domains by the number of tokens. The top two domain names are related to travel, and they account for about 10% of the total data.

We also applied Latent Dirichlet Allocation (LDA; Blei et al., 2003) to learn a distribution over latent topics in the extracted data, as this is a popular exploratory data analysis method. In LDA a topic is a unigram distribution over words, and each document is modeled as a distribution over topics. To create a set of documents from the extracted CommonCrawl data, we took the English side of the extracted parallel segments for each URL in the Spanish-English portion of the data. This gave us a total of 444,022 documents. In our first experiment, we used the MALLET toolkit (McCallum, 2002) to generate 20 topics, which are shown in Table 5.

Some of the topics that LDA finds correspond closely with specific domains, such as topics 1 (`blingee.com`) and 2 (`opensubtitles.org`). Several of the topics correspond to the travel domain. Foreign stop words appear in a few of the topics. Since our sys-

tem does not include any language identification, this is not surprising.<sup>5</sup> However it does suggest an avenue for possible improvement.

In our second LDA experiment, we compared our extracted CommonCrawl data with Europarl. We created a set of documents from both CommonCrawl and Europarl, and again used MALLET to generate 100 topics for this data.<sup>6</sup> We then labeled each document by its most likely topic (as determined by that topic’s mixture weights), and counted the number of documents from Europarl and CommonCrawl for which each topic was most prominent. While this is very rough, it gives some idea of where each topic is coming from. Table 6 shows a sample of these topics.

In addition to exploring topics in the datasets, we also performed additional intrinsic evaluation at the domain level, choosing top domains for three language pairs. We specifically classified sentence pairs as useful or boilerplate (Table 7). Among our observations, we find that commercial websites tend to contain less boilerplate material than encyclopedic websites, and that the ratios tend to be similar across languages in the same domain.

	FR	ES	DE
<code>www.booking.com</code>	52%	71%	52%
<code>www.hotel.info</code>	34%	44%	-
<code>memory-alpha.org</code>	34%	25%	55%

Table 7: Percentage of useful (non-boilerplate) sentences found by domain and language pair. `hotel.info` was not found in our German-English data.

## 4 Machine Translation Experiments

For our SMT experiments, we use the Moses toolkit (Koehn et al., 2007). In these experiments, a baseline system is trained on an existing parallel corpus, and the experimental system is trained on the baseline corpus plus the mined parallel data. In all experiments we include the target side of the mined parallel data in the language model, in order to distinguish whether results are due to influences from parallel or monolingual data.

<sup>5</sup>We used MALLET’s stop word removal, but that is only for English.

<sup>6</sup>Documents were created from Europarl by taking “SPEAKER” tags as document boundaries, giving us 208,431 documents total.

Genre	Domain	Pages	Segments	Source Tokens	Target Tokens
	<i>Total</i>	444K	5.67M	71.5M	67.5M
travel	www.booking.com	13.4K	424K	5.23M	5.14M
travel	www.hotel.info	9.05K	156K	1.93M	2.13M
government	www.fao.org	2.47K	60.4K	1.07M	896K
religious	scriptures.lds.org	7.04K	47.2K	889K	960K
political	www.amnesty.org	4.83K	38.1K	641K	548K

Table 4: The top five domains from the Spanish-English portion of the data. The domains are ranked by the combined number of source and target tokens.

Index	Most Likely Tokens
1	glitter graphics profile comments share love size girl friends happy blingee cute anime twilight sexy emo
2	subtitles online web users files rar movies prg akas dwls xvid dvdrip avi results download eng cd movie
3	miles hotels city search hotel home page list overview select tokyo discount destinations china japan
4	english language students details skype american university school languages words england british college
5	translation japanese english chinese dictionary french german spanish korean russian italian dutch
6	products services ni system power high software design technology control national applications industry
7	en de el instructions amd hyper riv saab kfreesbd poland user fr pln org wikimedia pl commons fran norway
8	information service travel services contact number time account card site credit company business terms
9	people time life day good years work make god give lot long world book today great year end things
10	show km map hotels de hotel beach spain san italy resort del mexico rome portugal home santa berlin la
11	rotary international world club korea foundation district business year global hong kong president ri
12	hotel reviews stay guest rooms service facilities room smoking submitted customers desk score united hour
13	free site blog views video download page google web nero internet http search news links category tv
14	casino game games play domaine ago days music online poker free video film sports golf live world tags bet
15	water food attribution health mango japan massage medical body baby natural yen commons traditional
16	file system windows server linux installation user files set debian version support program install type
17	united kingdom states america house london street park road city inn paris york st france home canada
18	km show map hotels hotel featured search station museum amsterdam airport centre home city rue germany
19	hotel room location staff good breakfast rooms friendly nice clean great excellent comfortable helpful
20	de la en le el hotel es het del und die il est der les des das du para

Table 5: A list of 20 topics generated using the MALLET toolkit (McCallum, 2002) and their most likely tokens.

#### 4.1 News Domain Translation

Our first set of experiments are based on systems built for the 2012 Workshop on Statistical Machine Translation (WMT) (Callison-Burch et al., 2012) using all available parallel and monolingual data for that task, aside from the French-English Gigaword. In these experiments, we use 5-gram language models when the target language is English or German, and 4-gram language models for French and Spanish. We tune model weights using minimum error rate training (MERT; Och, 2003) on the WMT 2008 test data. The results are given in Table 8. For all language pairs and both test sets (WMT 2011 and WMT 2012), we show an improvement of around 0.5 BLEU.

We also included the French-English Gigaword in separate experiments given in Table 9, and Table 10 compares the sizes of the datasets used. These results show that even on top of a different, larger parallel corpus mined from the web, adding CommonCrawl data still yields an improvement.

#### 4.2 Open Domain Translation

A substantial appeal of web-mined parallel data is that it might be suitable to translation of domains other than news, and our topic modeling analysis (§3.3) suggested that this might indeed be the case. We therefore performed an additional set of experiments for Spanish-English, but we include test sets from outside the news domain.

Europarl	CommonCrawl	Most Likely Tokens
9	2975	hair body skin products water massage treatment natural oil weight acid plant
2	4383	river mountain tour park tours de day chile valley ski argentina national peru la
8	10377	ford mercury dealer lincoln amsterdam site call responsible affiliates displayed
7048	675	market services european competition small public companies sector internal
9159	1359	time president people fact make case problem clear good put made years situation
13053	849	commission council european parliament member president states mr agreement
1660	5611	international rights human amnesty government death police court number torture
1617	4577	education training people cultural school students culture young information

Table 6: A sample of topics along with the number of Europarl and CommonCrawl documents where they are the most likely topic in the mixture. We include topics that are mostly found in Europarl or CommonCrawl, and some that are somewhat prominent in both.

WMT 11	FR-EN	EN-FR	ES-EN	EN-ES	EN-DE
<b>Baseline</b>	30.46	29.96	30.79	32.41	16.12
<b>+Web Data</b>	30.92	30.51	31.05	32.89	16.74
WMT 12	FR-EN	EN-FR	ES-EN	EN-ES	EN-DE
<b>Baseline</b>	29.25	27.92	32.80	32.83	16.61
<b>+Web Data</b>	29.82	28.22	33.39	33.41	17.30

Table 8: BLEU scores for several language pairs before and after adding the mined parallel data to systems trained on data from WMT data.

WMT 11	FR-EN	EN-FR
<b>Baseline</b>	30.96	30.69
<b>+Web Data</b>	31.24	31.17
WMT 12	FR-EN	EN-FR
<b>Baseline</b>	29.88	28.50
<b>+Web Data</b>	30.08	28.76

Table 9: BLEU scores for French-English and English-French before and after adding the mined parallel data to systems trained on data from WMT data including the French-English Gigaword (Callison-Burch et al., 2011).

For these experiments, we also include training data mined from Wikipedia using a simplified version of the sentence aligner described by Smith et al. (2010), in order to determine how the effect of such data compares with the effect of web-mined data. The baseline system was trained using only the Europarl corpus (Koehn, 2005) as parallel data, and all experiments use the same language model trained on the target sides of Europarl, the English side of all linked Spanish-English Wikipedia articles, and the English side of the mined CommonCrawl data. We use a 5-gram language model and tune using MERT (Och,

Corpus	EN-FR	EN-ES	EN-DE
<b>News Commentary</b>	2.99M	3.43M	3.39M
<b>Europarl</b>	50.3M	49.2M	47.9M
<b>United Nations</b>	316M	281M	-
<b>FR-EN Gigaword</b>	668M	-	-
<b>CommonCrawl</b>	121M	68.8M	88.4M

Table 10: The size (in English tokens) of the training corpora used in the SMT experiments from Tables 8 and 9 for each language pair.

2003) on the WMT 2009 test set.

Unfortunately, it is difficult to obtain meaningful results on some open domain test sets such as the Wikipedia dataset used by Smith et al. (2010). Wikipedia copied across the public internet, and we did not have a simple way to filter such data from our mined datasets.

We therefore considered two tests that were less likely to be problematic. The Tatoeba corpus (Tiedemann, 2009) is a collection of example sentences translated into many languages by volunteers. The front page of [tatoeba.org](http://tatoeba.org) was discovered by our URL matching heuristics, but we excluded any sentence pairs that were found in the CommonCrawl data from this test set.

The second dataset is a set of crowdsourced translation of Spanish speech transcriptions from the Spanish Fisher corpus.<sup>7</sup> As part of a research effort on cross-lingual speech applications, we obtained English translations of the data using Amazon Mechanical Turk, following a protocol similar to one described by Zaidan and Callison-Burch (2011): we provided clear instructions, employed several quality control measures, and obtained redundant translations of the complete dataset (Lopez et al., 2013). The advantage of this data for our open domain translation test is twofold. First, the Fisher dataset consists of conversations in various Spanish dialects on a wide variety of prompted topics. Second, because we obtained the translations ourselves, we could be absolutely assured that they did not appear in some form anywhere on the Web, making it an ideal blind test.

	<b>WMT10</b>	<b>Tatoeba</b>	<b>Fisher</b>
<b>Europarl</b>	89/72/46/20	94/75/45/18	87/69/39/13
<b>+Wiki</b>	92/78/52/24	96/80/50/21	91/75/44/15
<b>+Web</b>	96/82/56/27	99/88/58/26	96/83/51/19
<b>+Both</b>	96/84/58/29	99/89/60/27	96/83/52/20

Table 11:  $n$ -gram coverage percentages (up to 4-grams) of the source side of our test sets given our different parallel training corpora computed at the type level.

	<b>WMT10</b>	<b>Tatoeba</b>	<b>Fisher</b>
<b>Europarl</b>	27.21	36.13	46.32
<b>+Wiki</b>	28.03	37.82	49.34
<b>+Web</b>	28.50	41.07	51.13
<b>+Both</b>	28.74	41.12	52.23

Table 12: BLEU scores for Spanish-English before and after adding the mined parallel data to a baseline Europarl system.

We used 1000 sentences from each of the Tatoeba and Fisher datasets as test. For comparison, we also test on the WMT 2010 test set (Callison-Burch et al., 2010). Following Munteanu and Marcu (2005), we show the  $n$ -gram coverage of each corpus (percentage of  $n$ -grams from the test corpus which are also found in the training corpora) in Table 11. Table 12 gives end-to-end results, which show a strong improvement on the WMT test set (1.5 BLEU), and larger

<sup>7</sup>Linguistic Data Consortium LDC2010T04.

improvements on Tatoeba and Fisher (almost 5 BLEU).

## 5 Discussion

Web-mined parallel texts have been an exclusive resource of large companies for several years. However, when web-mined parallel text is available to everyone at little or no cost, there will be much greater potential for groundbreaking research to come from all corners. With the advent of public services such as Amazon Web Services and the Common Crawl, this may soon be a reality. As we have shown, it is possible to obtain parallel text for many language pairs in a variety of domains very cheaply and quickly, and in sufficient quantity and quality to improve statistical machine translation systems. However, our effort has merely scratched the surface of what is possible with this resource. We will make our code and data available so that others can build on these results.

Because our system is so simple, we believe that our results represent lower bounds on the gains that should be expected in performance of systems previously trained only on curated datasets. There are many possible means through which the system could be improved, including more sophisticated techniques for identifying matching URLs, better alignment, better language identification, better filtering of data, and better exploitation of resulting cross-domain datasets. Many of the components of our pipeline were basic, leaving considerable room for improvement. For example, the URL matching strategy could easily be improved for a given language pair by spending a little time crafting regular expressions tailored to some major websites. Callison-Burch et al. (2011) gathered almost 1 trillion tokens of French-English parallel data this way. Another strategy for mining parallel webpage pairs is to scan the HTML for links to the same page in another language (Nie et al., 1999).

Other, more sophisticated techniques may also be possible. Uszkoreit et al. (2010), for example, translated all non-English webpages into English using an existing translation system and used near-duplicate detection methods to find candidate parallel document pairs. Ture and Lin (2012) had a similar approach for finding parallel Wikipedia documents by using near-duplicate detection, though they did not need to apply a full translation system to all non-English documents.

Instead, they represented documents in bag-of-words vector space, and projected non-English document vectors into the English vector space using the translation probabilities of a word alignment model. By comparison, one appeal of our simple approach is that it requires only a table of language codes. However, with this system in place, we could obtain enough parallel data to bootstrap these more sophisticated approaches.

It is also compelling to consider ways in which web-mined data obtained from scratch could be used to bootstrap other mining approaches. For example, Smith et al. (2010) mine parallel sentences from *comparable* documents in Wikipedia, demonstrating substantial gains on open domain translation. However, their approach required seed parallel data to learn models used in a classifier. We imagine a two-step process, first obtaining parallel data from the web, followed by comparable data from sources such as Wikipedia using models bootstrapped from the web-mined data. Such a process could be used to build translation systems for new language pairs in a very short period of time, hence fulfilling one of the original promises of SMT.

## Acknowledgements

Thanks to Ann Irvine, Jonathan Weese, and our anonymous reviewers from NAACL and ACL for comments on previous drafts. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement 288487 (MosesCore). This research was partially funded by the Johns Hopkins University Human Language Technology Center of Excellence, and by gifts from Google and Microsoft.

## References

- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar F. Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and Metrics/MATR*, WMT '10, pages 17–53. Association for Computational Linguistics.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar F. Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 22–64. Association for Computational Linguistics.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, June. Association for Computational Linguistics.
- Jiang Chen and Jian-Yun Nie. 2000. Parallel web text mining for cross-language ir. In *IN IN PROC. OF RIAO*, pages 62–77.
- J. Dean and S. Ghemawat. 2004. Mapreduce: simplified data processing on large clusters. In *Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation-Volume 6*, pages 10–10. USENIX Association.
- William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Comput. Linguist.*, 19:75–102, March.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180. Association for Computational Linguistics.
- P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5.
- Edward Loper and Steven Bird. 2002. Nltk: the natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics - Volume 1*, ETMTNLP '02, pages 63–70. Association for Computational Linguistics.
- Adam Lopez, Matt Post, and Chris Callison-Burch. 2013. Parallel speech, transcription, and translation: The Fisher and Callhome Spanish-English speech translation corpora. Technical Report 11, Johns Hopkins University Human Language Technology Center of Excellence.
- Marco Lui and Timothy Baldwin. 2012. langid.py: an off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations*, ACL '12, pages 25–30. Association for Computational Linguistics.
- Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.

- Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *Comput. Linguist.*, 31:477–504, December.
- Jian-Yun Nie, Michel Simard, Pierre Isabelle, and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 74–81, New York, NY, USA. ACM.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *acl*, pages 160–167, Sapporo, Japan.
- P. Resnik and N. A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.
- Philip Resnik. 1999. Mining the web for bilingual text. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 527–534. Association for Computational Linguistics.
- Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting Parallel Sentences from Comparable Corpora using Document Level Alignment. In *NAACL 2010*.
- Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.
- Ferhan Ture and Jimmy Lin. 2012. Why not grab a free lunch? mining large corpora for parallel sentences to improve translation modeling. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 626–630, Montréal, Canada, June. Association for Computational Linguistics.
- Jakob Uszkoreit, Jay M. Ponte, Ashok C. Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1101–1109. Association for Computational Linguistics.
- Ashish Venugopal, Jakob Uszkoreit, David Talbot, Franz J. Och, and Juri Ganitkevitch. 2011. Watermarking the outputs of structured prediction with an application in statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1363–1372. Association for Computational Linguistics.
- Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *Proc. of ACL*.

# A Sentence Compression Based Framework to Query-Focused Multi-Document Summarization

Lu Wang<sup>1</sup> Hema Raghavan<sup>2</sup> Vittorio Castelli<sup>2</sup> Radu Florian<sup>2</sup> Claire Cardie<sup>1</sup>

<sup>1</sup>Department of Computer Science, Cornell University, Ithaca, NY 14853, USA

{luwang, cardie}@cs.cornell.edu

<sup>2</sup>IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

{hrahav, vittorio, raduf}@us.ibm.com

## Abstract

We consider the problem of using sentence compression techniques to facilitate query-focused multi-document summarization. We present a sentence-compression-based framework for the task, and design a series of learning-based compression models built on parse trees. An innovative beam search decoder is proposed to efficiently find highly probable compressions. Under this framework, we show how to integrate various indicative metrics such as linguistic motivation and query relevance into the compression process by deriving a novel formulation of a compression scoring function. Our best model achieves statistically significant improvement over the state-of-the-art systems on several metrics (e.g. 8.0% and 5.4% improvements in ROUGE-2 respectively) for the DUC 2006 and 2007 summarization task.

## 1 Introduction

The explosion of the Internet clearly warrants the development of techniques for organizing and presenting information to users in an effective way. Query-focused multi-document summarization (MDS) methods have been proposed as one such technique and have attracted significant attention in recent years. The goal of query-focused MDS is to synthesize a brief (often fixed-length) and well-organized summary from a set of topic-related documents that answer a complex question or address a topic statement. The resulting summaries, in turn, can support a number of information analysis applications including open-ended question answering, recommender systems, and summarization of search engine results. As further evidence of its importance, the Document Understanding Conference (DUC) has used query-focused MDS as its main task since 2004 to foster

new research on automatic summarization in the context of users' needs.

To date, most top-performing systems for multi-document summarization—whether query-specific or not—remain largely *extractive*: their summaries are comprised exclusively of sentences selected directly from the documents to be summarized (Erkan and Radev, 2004; Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tür, 2011). Despite their simplicity, extractive approaches have some disadvantages. First, lengthy sentences that are partly relevant are either excluded from the summary or (if selected) can block the selection of other important sentences, due to summary length constraints. In addition, when people write summaries, they tend to abstract the content and seldom use entire sentences taken verbatim from the original documents. In news articles, for example, most sentences are lengthy and contain both potentially useful information for a summary as well as unnecessary details that are better omitted. Consider the following DUC query as input for a MDS system:<sup>1</sup> “*In what ways have stolen artworks been recovered? How often are suspects arrested or prosecuted for the thefts?*” One manually generated summary includes the following sentence but removes the bracketed words in gray:

A man suspected of stealing a million-dollar collection of [hundreds of ancient] Nepalese and Tibetan art objects in New York [11 years ago] was arrested [Thursday at his South Los Angeles home, where he had been hiding the antiquities, police said].

In this example, the compressed sentence is rela-

<sup>1</sup>From DUC 2005, query for topic d422g.

tively more succinct and readable than the original (e.g. in terms of Flesch-Kincaid Reading Ease Score (Kincaid et al., 1975)). Likewise, removing information irrelevant to the query (e.g. “11 years ago”, “police said”) is crucial for query-focused MDS.

Sentence compression techniques (Knight and Marcu, 2000; Clarke and Lapata, 2008) are the standard for producing a compact and grammatical version of a sentence while preserving relevance, and prior research (e.g. Lin (2003)) has demonstrated their potential usefulness for generic document summarization. Similarly, strides have been made to incorporate sentence compression into query-focused MDS systems (Zajic et al., 2006). Most attempts, however, fail to produce better results than those of the best systems built on pure extraction-based approaches that use no sentence compression.

In this paper we investigate the role of sentence compression techniques for query-focused MDS. We extend existing work in the area first by investigating the role of *learning-based* sentence compression techniques. In addition, we design three types of approaches to sentence-compression—*rule-based*, *sequence-based* and *tree-based*—and examine them within our compression-based framework for query-specific MDS. Our top-performing sentence compression algorithm incorporates measures of query relevance, content importance, redundancy and language quality, among others. Our tree-based methods rely on a scoring function that allows for easy and flexible tailoring of sentence compression to the summarization task, ultimately resulting in significant improvements for MDS, while at the same time remaining competitive with existing methods in terms of sentence compression, as discussed next.

We evaluate the summarization models on the standard Document Understanding Conference (DUC) 2006 and 2007 corpora<sup>2</sup> for query-focused MDS and find that all of our compression-based summarization models achieve statistically significantly better performance than the best DUC 2006 systems. Our best-performing system yields an 11.02 ROUGE-2 score (Lin and Hovy, 2003), a 8.0% improvement over the best reported score (10.2 (Davis et al., 2012)) on the

<sup>2</sup>We believe that we can easily adapt our system for tasks (e.g. TAC-08’s opinion summarization or TAC-09’s update summarization) or domains (e.g. web pages or wikipedia pages). We reserve that for future work.

DUC 2006 dataset, and an 13.49 ROUGE-2, a 5.4% improvement over the best score in DUC 2007 (12.8 (Davis et al., 2012)). We also observe substantial improvements over previous systems w.r.t. the manual Pyramid (Nenkova and Passonneau, 2004) evaluation measure (26.4 vs. 22.9 (Jagarlamudi et al., 2006)); human annotators furthermore rate our system-generated summaries as having less redundancy and comparable quality w.r.t. other linguistic quality metrics. With these results we believe we are the first to successfully show that sentence compression can provide statistically significant improvements over pure extraction-based approaches for query-focused MDS.

## 2 Related Work

Existing research on query-focused multi-document summarization (MDS) largely relies on extractive approaches, where systems usually take as input a set of documents and select the top relevant sentences for inclusion in the final summary. A wide range of methods have been employed for this task. For unsupervised methods, sentence importance can be estimated by calculating topic signature words (Lin and Hovy, 2000; Conroy et al., 2006), combining query similarity and document centrality within a graph-based model (Otterbacher et al., 2005), or using a Bayesian model with sophisticated inference (Daumé and Marcu, 2006). Davis et al. (2012) first learn the term weights by Latent Semantic Analysis, and then greedily select sentences that cover the maximum combined weights. Supervised approaches have mainly focused on applying discriminative learning for ranking sentences (Fuentes et al., 2007). Lin and Bilmes (2011) use a class of carefully designed submodular functions to reward the diversity of the summaries and select sentences greedily.

Our work is more related to the less studied area of sentence compression as applied to (single) document summarization. Zajic et al. (2006) tackle the query-focused MDS problem using a compress-first strategy: they develop heuristics to generate multiple alternative compressions of all sentences in the original document; these then become the candidates for extraction. This approach, however, does not outperform some extraction-based approaches. A similar idea has been studied for MDS (Lin, 2003; Gillick and Favre, 2009),

but limited improvement is observed over extractive baselines with simple compression rules. Finally, although learning-based compression methods are promising (Martins and Smith, 2009; Berg-Kirkpatrick et al., 2011), it is unclear how well they handle issues of redundancy.

Our research is also inspired by probabilistic sentence-compression approaches, such as the noisy-channel model (Knight and Marcu, 2000; Turner and Charniak, 2005), and its extension via synchronous context-free grammars (SCFG) (Aho and Ullman, 1969; Lewis and Stearns, 1968) for robust probability estimation (Galley and McKeown, 2007). Rather than attempt to derive a new parse tree like Knight and Marcu (2000) and Galley and McKeown (2007), we learn to safely remove a set of constituents in our parse tree-based compression model while preserving grammatical structure and essential content. Sentence-level compression has also been examined via a discriminative model McDonald (2006), and Clarke and Lapata (2008) also incorporate discourse information by using integer linear programming.

### 3 The Framework

We now present our query-focused MDS framework consisting of three steps: Sentence Ranking, Sentence Compression and Post-processing. First, sentence ranking determines the importance of each sentence given the query. Then, a sentence compressor iteratively generates the most likely succinct versions of the ranked sentences, which are cumulatively added to the summary, until a length limit is reached. Finally, the post-processing stage applies coreference resolution and sentence reordering to build the summary.

**Sentence Ranking.** This stage aims to rank sentences in order of relevance to the query. Unsurprisingly, ranking algorithms have been successfully applied to this task. We experimented with two of them – Support Vector Regression (SVR) (Mozer et al., 1997) and LambdaMART (Burges et al., 2007). The former has been used previously for MDS (Ouyang et al., 2011). LambdaMart on the other hand has shown considerable success in information retrieval tasks (Burges, 2010); we are the first to apply it to summarization. For training, we use 40 topics (i.e. queries) from the DUC 2005 corpus (Dang, 2005) along with their manually generated abstracts. As in previous work (Shen and Li,

Basic Features
relative/absolute position
is among the first 1/3/5 sentences?
number of words (with/without stopwords)
number of words more than 5/10 (with/without stopwords)
Query-Relevant Features
unigram/bigram/skip bigram (at most four words apart) overlap
unigram/bigram TF/TF-IDF similarity
mention overlap
subject/object/indirect object overlap
semantic role overlap
relation overlap
Query-Independent Features
average/total unigram/bigram IDF/TF-IDF
unigram/bigram TF/TF-IDF similarity with the centroid of the cluster
average/sum of sumBasic/SumFocus (Toutanova et al., 2007)
average/sum of mutual information
average/sum of number of topic signature words (Lin and Hovy, 2000)
basic/improved sentence scorers from Conroy et al. (2006)
Content Features
contains verb/web link/phone number?
contains/portion of words between parentheses

Table 1: Sentence-level features for sentence ranking.

2011; Ouyang et al., 2011), we use the ROUGE-2 score, which measures bigram overlap between a sentence and the abstracts, as the objective for regression.

While space limitations preclude a longer discussion of the full feature set (ref. Table 1), we describe next the query-relevant features used for sentence ranking as these are the most important for our summarization setting. The goal of this feature subset is to determine the similarity between the query and each candidate sentence. When computing similarity, we remove stopwords as well as the words “discuss, describe, specify, explain, identify, include, involve, note” that are adopted and extended from Conroy et al. (2006). Then we conduct simple query expansion based on the title of the topic and cross-document coreference resolution. Specifically, we first add the words from the topic title to the query. And for each mention in the query, we add other mentions within the set of documents that corefer with this mention. Finally, we compute two versions of the features—one based on the original query and another on the expanded one. We also derive the semantic role overlap and relation instance overlap between the query and each sentence. Cross-document coreference resolution, semantic role labeling and relation extraction are accomplished via the methods described in Section 5.

**Sentence Compression.** As the main focus of this paper, we propose three types of compression methods, described in detail in Section 4 below.

**Post-processing.** Post-processing performs *coreference resolution* and *sentence ordering*.

Basic Features	Syntactic Tree Features
first 1/3/5 tokens (toks)? last 1/3/5 toks? first letter/all letters capitalized? is negation? is stopword? <b>Dependency Tree Features</b> dependency relation (dep rel) parent/grandparent dep rel is the root? has a depth larger than 3/5?	POS tag parent/grandparent label leftmost child of parent? second leftmost child of parent? is headword? in NP/VP/ADVP/ADJP chunk? <b>Semantic Features</b> is a predicate? semantic role label
Rule-Based Features	
For each rule in Table 2, we construct a corresponding feature to indicate whether the token is identified by the rule.	

Table 3: Token-level features for sequence-based compression.

We replace each pronoun with its referent unless they appear in the same sentence. For sentence ordering, each compressed sentence is assigned to the most similar (tf-idf) query sentence. Then a Chronological Ordering algorithm (Barzilay et al., 2002) sorts the sentences for each query based first on the time stamp, and then the position in the source document.

## 4 Sentence Compression

Sentence compression is typically formulated as the problem of removing secondary information from a sentence while maintaining its grammaticality and semantic structure (Knight and Marcu, 2000; McDonald, 2006; Galley and McKeown, 2007; Clarke and Lapata, 2008). We leave other rewrite operations, such as paraphrasing and re-ordering, for future work. Below we describe the sentence compression approaches developed in this research: RULE-BASED COMPRESSION, SEQUENCE-BASED COMPRESSION, and TREE-BASED COMPRESSION.

### 4.1 Rule-based Compression

Turner and Charniak (2005) have shown that applying hand-crafted rules for trimming sentences can improve both content and linguistic quality. Our rule-based approach extends existing work (Conroy et al., 2006; Toutanova et al., 2007) to create the linguistically-motivated compression rules of Table 2. To avoid ill-formed output, we disallow compressions of more than 10 words by each rule.

### 4.2 Sequence-based Compression

As in McDonald (2006) and Clarke and Lapata (2008), our sequence-based compression model makes a binary “keep-or-delete” decision for each word in the sentence. In contrast, however, we

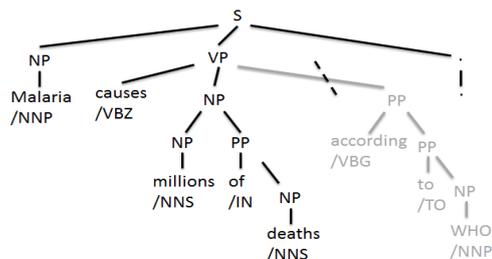


Figure 1: Diagram of tree-based compression. The nodes to be dropped are grayed out. In this example, the root of the gray subtree (a “PP”) would be labeled REMOVE. Its siblings and parent are labeled RETAIN and PARTIAL, respectively. The trimmed tree is realized as “Malaria causes millions of deaths.”

view compression as a sequential tagging problem and make use of linear-chain Conditional Random Fields (CRFs) (Lafferty et al., 2001) to select the most likely compression. We represent each sentence as a sequence of tokens,  $X = x_0x_1 \dots x_n$ , and generate a sequence of labels,  $Y = y_0y_1 \dots y_n$ , that encode which tokens are kept, using a BIO label format: {B-RETAIN denotes the beginning of a retained sequence, I-RETAIN indicates tokens “inside” the retained sequence, O marks tokens to be removed}.

The CRF model is built using the features shown in Table 3. “Dependency Tree Features” encode the grammatical relations in which each word is involved as a dependent. For the “Syntactic Tree”, “Dependency Tree” and “Rule-Based” features, we also include features for the two words that precede and the two that follow the current word. Detailed descriptions of the training data and experimental setup are in Section 5.

During inference, we find the maximally likely sequence  $Y$  according to a CRF with parameter  $\theta$  ( $Y = \arg \max_{Y'} P(Y'|X; \theta)$ ), while simultaneously enforcing the rules of Table 2 to reduce the hypothesis space and encourage grammatical compression. To do this, we encode these rules as features for each token, and whenever these feature functions fire, we restrict the possible label for that token to “O”.

### 4.3 Tree-based Compression

Our tree-based compression methods are in line with syntax-driven approaches (Galley and McKeown, 2007), where operations are carried out on parse tree constituents. Unlike previous work (Knight and Marcu, 2000; Galley and McKeown, 2007), we do not produce a new parse tree,

Rule	Example
Header	[MOSCOW , October 19 ( Xinhua ) –] Russian federal troops Tuesday continued...
Relative dates	...Centers for Disease Control confirmed [Tuesday] that there was...
Intra-sentential attribution	...fueling the La Nina weather phenomenon, [the U.N. weather agency said].
Lead adverbials	[Interestingly], while the Democrats tend to talk about...
Noun appositives	Wayne County Prosecutor [John O’Hara] wanted to send a message...
Nonrestrictive relative clause	Putin, [who was born on October 7, 1952 in Leningrad], was elected in the presidential election...
Adverbial clausal modifiers (Lead sentence)	[Starting in 1998], California will require 2 per cent of a manufacturer... [Given the short time], car makers see electric vehicles as...
Within Parentheses	...to Christian home schoolers in the early 1990s [(www.homecomputermarket.com)].

Table 2: Linguistically-motivated rules for sentence compression. The grayed-out words in brackets are removed.

but focus on learning to identify the proper set of constituents to be removed. In particular, when a node is dropped from the tree, all words it subsumes will be deleted from the sentence.

Formally, given a parse tree  $T$  of the sentence to be compressed and a tree traversal algorithm,  $T$  can be presented as a list of ordered constituent nodes,  $T = t_0 t_1 \dots t_m$ . Our objective is to find a set of labels,  $L = l_0 l_1 \dots l_m$ , where  $l_i \in \{\text{RETAIN, REMOVE, PARTIAL}\}$ . RETAIN (RET) and REMOVE (REM) denote whether the node  $t_i$  is retained or removed. PARTIAL (PAR) means  $t_i$  is partly removed, i.e. at least one child subtree of  $t_i$  is dropped.

Labels are identified, in order, according to the tree traversal algorithm. Every node label needs to be *compatible* with the labeling history: given a node  $t_i$ , and a set of labels  $l_0 \dots l_{i-1}$  predicted for nodes  $t_0 \dots t_{i-1}$ ,  $l_i = \text{RET}$  or  $l_i = \text{REM}$  is *compatible* with the history when all children of  $t_i$  are labeled as RET or REM, respectively;  $l_i = \text{PAR}$  is *compatible* when  $t_i$  has at least two descendents  $t_j$  and  $t_k$  ( $j < i$  and  $k < i$ ), one of which is RETAINED and the other, REMOVED. As such, the root of the gray subtree in Figure 1 is labeled as REM; its left siblings as RET; its parent as PAR.

As the space of possible compressions is exponential in the number of leaves in the parse tree, instead of looking for the globally optimal solution, we use beam search to find a set of highly likely compressions and employ a language model trained on a large corpus for evaluation.

**A Beam Search Decoder.** The beam search decoder (see Algorithm 1) takes as input the sentence’s parse tree  $T = t_0 t_1 \dots t_m$ , an ordering  $O$  for traversing  $T$  (e.g. postorder) as a sequence of nodes in  $T$ , the set  $L$  of possible node labels, a scoring function  $S$  for evaluating each sentence compression hypothesis, and a beam size  $N$ . Specifically,  $O$  is a permutation on the set  $\{0, 1, \dots, m\}$ —each element an

index onto  $T$ . Following  $O$ ,  $T$  is re-ordered as  $t_{O_0} t_{O_1} \dots t_{O_m}$ , and the decoder considers each ordered constituent  $t_{O_i}$  in turn. In iteration  $i$ , all existing sentence compression hypotheses are expanded by one node,  $t_{O_i}$ , labeling it with **all compatible** labels. The new hypotheses (usually sub-sentences) are ranked by the scorer  $S$  and the top  $N$  are preserved to be extended in the next iteration. See Figure 2 for an example.

```

Input : parse tree  $T$ , ordering  $O = O_0 O_1 \dots O_m$ ,
          $L = \{\text{RET, REM, PAR}\}$ , hypothesis scorer  $S$ ,
         beam size  $N$ 
Output:  $N$  best compressions

stack  $\leftarrow \Phi$  (empty set);
foreach node  $t_{O_i}$  in  $T = t_{O_0} \dots t_{O_m}$  do
  if  $i == 0$  (first node visited) then
    foreach label  $l_{O_0}$  in  $L$  do
      newHypothesis  $h' \leftarrow [l_{O_0}]$ ;
      put  $h'$  into Stack;
    end
  else
    newStack  $\leftarrow \Phi$  (empty set);
    foreach hypothesis  $h$  in stack do
      foreach label  $l_{O_i}$  in  $L$  do
        if  $l_{O_i}$  is compatible then
          newHypothesis  $h' \leftarrow h + [l_{O_i}]$ ;
          put  $h'$  into newStack;
        end
      end
    end
    stack  $\leftarrow$  newStack;
  end
Apply  $S$  to sort hypotheses in stack in descending order;
Keep the  $N$  best hypotheses in stack;
end

```

Algorithm 1: Beam search decoder.

Our **BASIC Tree-based Compression** instantiates the beam search decoder with postorder traversal and a hypothesis scorer that takes a possible sentence compression—a sequence of nodes (e.g.  $t_{O_0} \dots t_{O_k}$ ) and their labels (e.g.  $l_{O_0} \dots l_{O_k}$ )—and returns  $\sum_{j=1}^k \log P(l_{O_j} | t_{O_j})$  (denoted later as  $Score_{Basic}$ ). The probability is estimated by a Maximum Entropy classifier (Berger et al.,

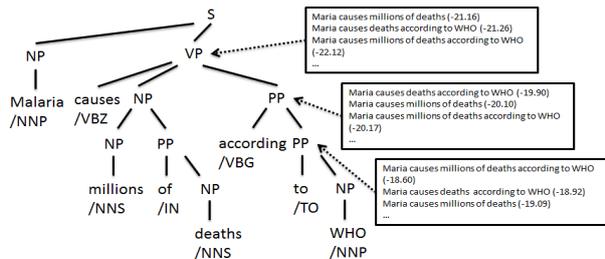


Figure 2: Example of beam search decoding. For postorder traversal, the three nodes are visited in a bottom-up order. The associated compression hypotheses (boxed) are ranked based on the scores in parentheses. Beam scores for other nodes are omitted.

Basic Features	Syntactic Tree Features
projection falls w/in first 1/3/5 toks?*	constituent label
projection falls w/in last 1/3/5 toks?*	parent left/right sibling label
subsumes first 1/3/5 toks?*	grandparent left/right sibling label
subsumes last 1/3/5 toks?*	is leftmost child of parent?
number of words larger than 5/10?*	is second leftmost child of parent?
is leaf node?*	is head node of parent?
is root of parsing tree?*	label of its head node
has word with first letter capitalized?	has a depth greater than 3/5/10?
has word with all letters capitalized?	<b>Dependency Tree Features</b>
has negation?	dep rel of head node†
has stopwords?	dep rel of parent's head node†
<b>Semantic Features</b>	dep rel of grandparent's head node†
the head node has predicate?	contain root of dep tree?†
semantic roles of head node	has a depth larger than 3/5?†
<b>Rule-Based Features</b>	
For each rule in Table 2, we construct a corresponding feature to indicate whether the token is identified by the rule.	

Table 4: Constituent-level features for tree-based compression. \* or † denote features that are concatenated with every Syntactic Tree feature to compose a new one.

1996) trained at the constituent level using the features in Table 4. We also apply the rules of Table 2 during the decoding process. Concretely, if the words subsumed by a node are identified by any rule, we only consider REM as the node's label.

Given the  $N$ -best compressions from the decoder, we evaluate the yield of the trimmed trees using a language model trained on the Gigaword (Graff, 2003) corpus and return the compression with the highest probability. Thus, the decoder is quite flexible — its learned scoring function allows us to incorporate features salient for sentence compression while its language model guarantees the linguistic quality of the compressed string. In the sections below we consider additional improvements.

### 4.3.1 Improving Beam Search

**CONTEXT-aware search** is based on the intuition that predictions on preceding context can be leveraged to facilitate the prediction of the current node. For example, parent nodes with

children that have all been removed (retained) should have a label of REM (RET). In light of this, we encode these contextual predictions as additional features of  $S$ , that is, ALL-CHILDREN-REMOVED/RETAINED, ANY-LEFTSIBLING-REMOVED/RETAINED/PARTLY\_REMOVED, LABEL-OF-LEFT-SIBLING/HEAD-NODE.

**HEAD-driven search** modifies the BASIC postorder tree traversal by visiting the head node first at each level, leaving other orders unchanged. In a nutshell, if the head node is dropped, then its modifiers need not be preserved. We adopt the same features as CONTEXT-aware search, but remove those involving left siblings. We also add one more feature: LABEL-OF-THE-HEAD-NODE-IT-MODIFIES.

### 4.3.2 Task-Specific Sentence Compression

The current scorer  $Score_{Basic}$  is still fairly naive in that it focuses only on features of the sentence to be compressed. *However extra-sentential knowledge can also be important for query-focused MDS.* For example, information regarding relevance to the query might lead the decoder to produce compressions better suited for the summary. Towards this goal, we construct a compression scoring function—the *multi-scorer* (MULTI)—that allows the incorporation of multiple task-specific scorers. Given a hypothesis at any stage of decoding, which yields a sequence of words  $W = w_0w_1\dots w_j$ , we propose the following component scorers.

**Query Relevance.** Query information ought to guide the compressor to identify the relevant content. The query  $Q$  is expanded as described in Section 3. Let  $|W \cap Q|$  denote the number of unique overlapping words between  $W$  and  $Q$ , then  $score_q = |W \cap Q|/|W|$ .

**Importance.** A query-independent importance score is defined as the average Sum-Basic (Toutanova et al., 2007) value in  $W$ , i.e.  $score_{im} = \sum_{i=1}^j SumBasic(w_i)/|W|$ .

**Language Model.** We let  $score_{lm}$  be the probability of  $W$  computed by a language model.

**Cross-Sentence Redundancy.** To encourage diversified content, we define a redundancy score to discount replicated content:  $score_{red} = 1 - |W \cap C|/|W|$ , where  $C$  is the words already selected for the summary.

The *multi-scorer* is defined as a linear combination of the component scorers: Let  $\vec{\alpha} = (\alpha_0, \dots, \alpha_4)$ ,  $0 \leq \alpha_i \leq 1$ ,  $\overrightarrow{score} = (score_{Basic}, score_q, score_{im}, score_{lm}, score_{red})$ ,

$$S = score_{multi} = \vec{\alpha} \cdot \overrightarrow{score} \quad (1)$$

The parameters  $\vec{\alpha}$  are tuned on a held-out tuning set by grid search. We linearly normalize the score of each metric, where the minimum and maximum values are estimated from the tuning data.

## 5 Experimental Setup

We evaluate our methods on the DUC 2005, 2006 and 2007 datasets (Dang, 2005; Dang, 2006; Dang, 2007), each of which is a collection of newswire articles. 50 complex queries (topics) are provided for DUC 2005 and 2006, 35 are collected for DUC 2007 main task. Relevant documents for each query are provided along with 4 to 9 human MDS abstracts. The task is to generate a summary within 250 words to address the query. We split DUC 2005 into two parts: 40 topics to train the sentence ranking models, and 10 for ranking algorithm selection and parameter tuning for the multi-scorer. DUC 2006 and DUC 2007 are reserved as held out test sets.

*Sentence Compression.* The dataset from Clarke and Lapata (2008) is used to train the CRF and MaxEnt classifiers (Section 4). It includes 82 newswire articles with one manually produced compression aligned to each sentence.

*Preprocessing.* Documents are processed by a full NLP pipeline, including token and sentence segmentation, parsing, semantic role labeling, and an information extraction pipeline consisting of mention detection, NP coreference, cross-document resolution, and relation detection (Florin et al., 2004; Luo et al., 2004; Luo and Zitouni, 2005).

*Learning for Sentence Ranking and Compression.* We use Weka (Hall et al., 2009) to train a support vector regressor and experiment with various rankers in RankLib (Dang, 2011)<sup>3</sup>. As LambdaMART has an edge over other rankers on the held-out dataset, we selected it to produce ranked sentences for further processing. For sequence-based compression using CRFs, we employ Mallet (McCallum, 2002) and integrate the Table 2 rules during inference. NLTK (Bird et al., 2009)

<sup>3</sup>Default parameters are used. If an algorithm needs a validation set, we use 10 out of 40 topics.

MaxEnt classifiers are used for tree-based compression. Beam size is fixed at 2000.<sup>4</sup> Sentence compressions are evaluated by a 5-gram language model trained on Gigaword (Graff, 2003) by SRILM (Stolcke, 2002).

## 6 Results

The results in Table 5 use the official ROUGE software with standard options<sup>5</sup> and report ROUGE-2 (R-2) (measures bigram overlap) and ROUGE-SU4 (R-SU4) (measures unigram and skip-bigram separated by up to four words). We compare our sentence-compression-based methods to the best performing systems based on ROUGE in DUC 2006 and 2007 (Jagarlamudi et al., 2006; Pingali et al., 2007), system by Davis et al. (2012) that report the best R-2 score on DUC 2006 and 2007 thus far, and to the purely extractive methods of SVR and LambdaMART.

Our sentence-compression-based systems (marked with †) show statistically significant improvements over pure extractive summarization for both R-2 and R-SU4 (paired *t*-test,  $p < 0.01$ ). This means our systems can effectively remove redundancy within the summary through compression. Furthermore, our HEAD-driven beam search method with MULTI-scorer beats all systems on DUC 2006<sup>6</sup> and all systems on DUC 2007 except the best system in terms of R-2 ( $p < 0.01$ ). Its R-SU4 score is also significantly ( $p < 0.01$ ) better than extractive methods, rule-based and sequence-based compression methods on both DUC 2006 and 2007. Moreover, our systems with learning-based compression have considerable compression rates, indicating their capability to remove superfluous words as well as improve summary quality.

**Human Evaluation.** The Pyramid (Nenkova and Passonneau, 2004) evaluation was developed to manually assess how many relevant facts or Summarization Content Units (SCUs) are captured by system summaries. We ask a professional annotator (who is not one of the authors, is highly experienced in annotating for various NLP tasks, and is fluent in English) to carry out a Pyramid evaluation on 10 randomly selected topics from

<sup>4</sup>We looked at various beam sizes on the heldout data, and observed that the performance peaks around this value.

<sup>5</sup>ROUGE-1.5.5.pl -n 4 -w 1.2 -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -a -d

<sup>6</sup>The system output from Davis et al. (2012) is not available, so significance tests are not conducted on it.

System	DUC 2006			DUC 2007		
	C Rate	R-2	R-SU4	C Rate	R-2	R-SU4
Best DUC system	–	9.56	15.53	–	12.62	17.90
Davis et al. (2012)	–	10.2	15.2	–	12.8	17.5
SVR	100%	7.78	13.02	100%	9.53	14.69
LambdaMART	100%	9.84	14.63	100%	12.34	15.62
Rule-based	78.99%	10.62 *†	15.73 †	78.11%	13.18†	18.15†
Sequence	76.34%	10.49 †	15.60 †	77.20%	13.25†	18.23†
Tree (BASIC + <i>Score<sub>Basic</sub></i> )	70.48%	10.49 †	15.86 †	69.27%	13.00†	18.29†
Tree (CONTEXT + <i>Score<sub>Basic</sub></i> )	65.21%	10.55 *†	16.10 †	63.44%	12.75	18.07†
Tree (HEAD + <i>Score<sub>Basic</sub></i> )	66.70%	10.66 *†	16.18 †	65.05%	12.93	18.15†
Tree (HEAD + MULTI)	70.20%	<b>11.02</b> *†	<b>16.25</b> †	73.40%	<b>13.49</b> †	<b>18.46</b> †

Table 5: Query-focused MDS performance comparison: C Rate or *compression rate* is the proportion of words preserved. R-2 (ROUGE-2) and R-SU4 (ROUGE-SU4) scores are multiplied by 100. “–” indicates that data is unavailable. BASIC, CONTEXT and HEAD represent the basic beam search decoder, context-aware and head-driven search extensions respectively. *Score<sub>Basic</sub>* and MULTI refer to the type of scorer used. Statistically significant improvements ( $p < 0.01$ ) over the best system in DUC 06 and 07 are marked with \*. † indicates statistical significance ( $p < 0.01$ ) over extractive approaches (SVR or LambdaMART). HEAD + MULTI outperforms all the other extract- and compression-based systems in R-2.

System	Pyr	Gra	Non-Red	Ref	Foc	Coh
Best DUC system (ROUGE)	22.9±8.2	3.5±0.9	3.5±1.0	3.5±1.1	3.6±1.0	2.9±1.1
Best DUC system (LQ)	–	4.0±0.8	4.2±0.7	3.8±0.7	3.6±0.9	3.4±0.9
Our System	<b>26.4±10.3</b>	3.0±0.9	4.0±1.1	3.6±1.0	3.4±0.9	2.8±1.0

Table 6: Human evaluation on our multi-scorer based system, Jagarlamudi et al. (2006) (Best DUC system (ROUGE)), and Lacatusu et al. (2006) (Best DUC system (LQ)). Our system can synthesize more relevant content according to Pyramid ( $\times 100$ ). We also examine linguistic quality (LQ) in Grammaticality (Gra), Non-redundancy (Non-Red), Referential clarity (Ref), Focus (Foc), and Structure and Coherence (Coh) like Dang (2006), each rated from 1 (very poor) to 5 (very good). Our system has better non-redundancy than Jagarlamudi et al. (2006) and is comparable to Jagarlamudi et al. (2006) and Lacatusu et al. (2006) in other metrics except grammaticality.

the DUC 2006 task with gold-standard SCU annotation in abstracts. The Pyramid score (see Table 6) is re-calculated for the system with best ROUGE scores in DUC 2006 (Jagarlamudi et al., 2006) along with our system by the same annotator to make a meaningful comparison.

We further evaluate the linguistic quality (LQ) of the summaries for the same 10 topics in accordance with the measurement in Dang (2006). Four native speakers who are undergraduate students in computer science (none are authors) performed the task. We compare our system based on HEAD-driven beam search with MULTI-scorer to the best systems in DUC 2006 achieving top ROUGE scores (Jagarlamudi et al., 2006) (Best DUC system (ROUGE)) and top linguistic quality scores (Lacatusu et al., 2006) (Best DUC system (LQ))<sup>7</sup>. The average score and standard deviation for each metric is displayed in Table 6. Our system achieves a higher Pyramid score, an indication that it captures more of the salient facts. We also

attain better non-redundancy than Jagarlamudi et al. (2006), meaning that human raters perceive less replicative content in our summaries. Scores for other metrics are comparable to Jagarlamudi et al. (2006) and Lacatusu et al. (2006), which either uses minimal non-learning-based compression rules or is a pure extractive system. However, our compression system sometimes generates less grammatical sentences, and those are mostly due to parsing errors. For example, parsing a clause starting with a past tense verb as an adverbial clausal modifier can lead to an ill-formed compression. Those issues can be addressed by analyzing  $k$ -best parse trees and we leave it in the future work. A sample summary from our multi-scorer based system is in Figure 3.

**Sentence Compression Evaluation.** We also evaluate sentence compression separately on (Clarke and Lapata, 2008), adopting the same partitions as (Martins and Smith, 2009), i.e. 1, 188 sentences for training and 441 for testing. Our compression models are compared with Hedge Trimmer (Dorr et al., 2003), a discriminative model proposed by McDonald (2006) and a

<sup>7</sup>Lacatusu et al. (2006) obtain the best scores in three linguistic quality metrics (i.e. grammaticality, focus, structure and coherence), and overall responsiveness on DUC 2006.

System	C Rate	Uni-Prec	Uni-Rec	Uni-F1	Rel-F1
HedgeTrimmer	57.64%	0.72	0.65	0.64	0.50
McDonald (2006)	70.95%	0.77	0.78	<i>0.77</i>	0.55
Martins and Smith (2009)	71.35%	0.77	0.78	<i>0.77</i>	0.56
Rule-based	87.65%	0.74	0.91	0.80	0.63
Sequence	70.79%	0.77	0.80	<i>0.76</i>	0.58
Tree (BASIC)	69.65%	0.77	0.79	0.75	0.56
Tree (CONTEXT)	67.01%	<b>0.79</b>	0.78	<i>0.76</i>	0.57
Tree (HEAD)	68.06%	<b>0.79</b>	0.80	<i>0.77</i>	<b>0.59</b>

Table 7: Sentence compression comparison. The true c rate is 69.06% for the test set. Tree-based approaches all use single-scorer. Our context-aware and head-driven tree-based approaches outperform all the other systems significantly ( $p < 0.01$ ) in precision (**Uni-Prec**) without sacrificing the recalls (i.e. there is no statistically significant difference between our models and McDonald (2006) / M & S (2009) with  $p > 0.05$ ). *Italicized* numbers for unigram F1 (**Uni-F1**) are statistically indistinguishable ( $p > 0.05$ ). Our head-driven tree-based approach also produces significantly better grammatical relations F1 scores (**Rel-F1**) than all the other systems except the rule-based method ( $p < 0.01$ ).

Topic <i>D0626H</i> : How were the bombings of the US embassies in Kenya and Tanzania conducted? What terrorist groups and individuals were responsible? How and where were the attacks planned?
WASHINGTON, August 13 (Xinhua) – President Bill Clinton Thursday condemned terrorist bomb attacks at U.S. embassies in Kenya and Tanzania and vowed to find the bombers and bring them to justice. Clinton met with his top aides Wednesday in the White House to assess the situation following the twin bombings at U.S. embassies in Kenya and Tanzania, which have killed more than 250 people and injured over 5,000, most of them Kenyans and Tanzanians. Local sources said the plan to bomb U.S. embassies in Kenya and Tanzania took three months to complete and bombers destined for Kenya were dispatched through Somali and Rwanda. FBI Director Louis Freeh, Attorney General Janet Reno and other senior U.S. government officials will hold a news conference at 1 p.m. EDT (1700GMT) at FBI headquarters in Washington “to announce developments in the investigation of the bombings of the U.S. embassies in Kenya and Tanzania,” the FBI said in a statement. ...

Figure 3: Part of the summary generated by the multi-scorer based summarizer for topic *D0626H* (DUC 2006). Grayed out words are removed. Query-irrelevant phrases, such as temporal information or source of the news, have been removed.

dependency-tree based compressor (Martins and Smith, 2009)<sup>8</sup>. We adopt the metrics in Martins and Smith (2009) to measure the unigram-level macro precision, recall, and F1-measure with respect to human annotated compression. In addition, we also compute the F1 scores of grammatical relations which are annotated by RASP (Briscoe and Carroll, 2002) according to Clarke and Lapata (2008).

In Table 7, our context-aware and head-driven tree-based compression systems show statistically significantly ( $p < 0.01$ ) higher precisions (**Uni-**

**Prec**) than all the other systems, without decreasing the recalls (**Uni-Rec**) significantly ( $p > 0.05$ ) based on a paired *t*-test. Unigram F1 scores (**Uni-F1**) in italics indicate that the corresponding systems are not statistically distinguishable ( $p > 0.05$ ). For grammatical relation evaluation, our head-driven tree-based system obtains statistically significantly ( $p < 0.01$ ) better F1 score (**Rel-F1**) than all the other systems except the rule-based system).

## 7 Conclusion

We have presented a framework for query-focused multi-document summarization based on sentence compression. We propose three types of compression approaches. Our tree-based compression method can easily incorporate measures of query relevance, content importance, redundancy and language quality into the compression process. By testing on a standard dataset using the automatic metric ROUGE, our models show substantial improvement over pure extraction-based methods and state-of-the-art systems. Our best system also yields better results for human evaluation based on Pyramid and achieves comparable linguistic quality scores.

## Acknowledgments

This work was supported in part by National Science Foundation Grant IIS-0968450 and a gift from Boeing. We thank Ding-Jung Han, Young-Suk Lee, Xiaoqiang Luo, Sameer Maskey, Myle Ott, Salim Roukos, Yiye Ruan, Ming Tan, Todd Ward, Bowen Zhou, and the ACL reviewers for valuable suggestions and advice on various aspects of this work.

<sup>8</sup>Thanks to André F.T. Martins for system outputs.

## References

- Alfred V. Aho and Jeffrey D. Ullman. 1969. Syntax directed translations and the pushdown assembler. *J. Comput. Syst. Sci.*, 3(1):37–56.
- Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *J. Artif. Int. Res.*, 17(1):35–55, August.
- Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. ACL '11, pages 481–490, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March.
- Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.
- T. Briscoe and J. Carroll. 2002. Robust accurate statistical annotation of general text.
- Christopher J.C. Burges, Robert Ragno, and Quoc Viet Le. 2007. Learning to rank with nonsmooth cost functions. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 193–200. MIT Press, Cambridge, MA.
- Christopher J. C. Burges. 2010. From RankNet to LambdaRank to LambdaMART: An overview. Technical report, Microsoft Research.
- Asli Celikyilmaz and Dilek Hakkani-Tür. 2011. Discovery of topically coherent sentences for extractive summarization. ACL '11, pages 491–499, Stroudsburg, PA, USA. Association for Computational Linguistics.
- James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: an integer linear programming approach. *J. Artif. Int. Res.*, 31(1):399–429, March.
- John M. Conroy, Judith D. Schlesinger, Dianne P. O'Leary, and Jade Goldstein, 2006. *Back to Basics: CLASSY 2006*. U.S. National Inst. of Standards and Technology.
- Hoa T. Dang. 2005. Overview of DUC 2005. In *Document Understanding Conference*.
- Hoa Tran Dang. 2006. Overview of DUC 2006. In *Proc. Document Understanding Workshop*, page 10 pages. NIST.
- Hoa T. Dang. 2007. Overview of DUC 2007. In *Document Understanding Conference*.
- Van Dang. 2011. RankLib. Online.
- Hal Daumé, III and Daniel Marcu. 2006. Bayesian query-focused summarization. ACL '06, pages 305–312, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Sashka T. Davis, John M. Conroy, and Judith D. Schlesinger. 2012. Occams - an optimal combinatorial covering algorithm for multi-document summarization. In *ICDM Workshops*, pages 454–463.
- Bonnie J Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: a parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop - Volume 5*, HLT-NAACL-DUC '03, pages 1 – 8, Stroudsburg, PA, USA. Association for Computational Linguistics, Association for Computational Linguistics.
- Günes Erkan and Dragomir R. Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479, December.
- Radu Florian, Hany Hassan, Abraham Ittycheriah, Hongyan Jing, Nanda Kambhatla, Xiaoqiang Luo, Nicolas Nicolov, and Salim Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *HLT-NAACL*, pages 1–8.
- Maria Fuentes, Enrique Alfonseca, and Horacio Rodríguez. 2007. Support vector machines for query-focused summarization trained and evaluated on pyramid data. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 57–60, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. NAACL '07, pages 180–187, Rochester, New York, April. Association for Computational Linguistics.
- Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, ILP '09, pages 10–18, Stroudsburg, PA, USA. Association for Computational Linguistics.
- David Graff. 2003. English Gigaword.
- Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. NAACL '09, pages 362–370, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: an update. *SIGKDD Explorer Newsl.*, 11(1):10–18, November.
- Jagadeesh Jagarlamudi, Prasad Pingali, and Vasudeva Varma, 2006. *Query Independent Sentence Scoring approach to DUC 2006*.
- J. Peter Kincaid, Robert P. Fishburne, Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel. Technical report, February.
- Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization - step one: Sentence compression. AAI '00, pages 703–710. AAI Press.
- Finley Lacatusu, Andrew Hickl, Kirk Roberts, Ying Shi, Jeremy Bensley, Bryan Rink, Patrick Wang, and Lara Taylor, 2006. *LCC's gistexter at duc 2006: Multi-strategy multi-document summarization*.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In

- Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- P. M. Lewis, II and R. E. Stearns. 1968. Syntax-directed transduction. *J. ACM*, 15(3):465–488, July.
- Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 510–520, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on Computational linguistics - Volume 1*, COLING '00, pages 495–501, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 71–78.
- Chin-Yew Lin. 2003. Improving summarization performance by sentence compression: a pilot study. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages - Volume 11*, AsianIR '03, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Xiaoqiang Luo and Imed Zitouni. 2005. Multi-lingual coreference resolution with syntactic features. In *HLT/EMNLP*.
- Xiaoqiang Luo, Abraham Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *ACL*, pages 135–142.
- André F. T. Martins and Noah A. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, ILP '09, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- Ryan McDonald. 2006. Discriminative Sentence Compression with Soft Syntactic Constraints. In *Proceedings of the 11th EACL*, Trento, Italy, April.
- Michael Mozer, Michael I. Jordan, and Thomas Petsche, editors. 1997. *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*. MIT Press.
- Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 145–152, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Jahna Otterbacher, Güneş Erkan, and Dragomir R. Radev. 2005. Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 915–922, Stroudsburg, PA, USA. Association for Computational Linguistics.
- You Ouyang, Wenjie Li, Sujian Li, and Qin Lu. 2011. Applying regression models to query-focused multi-document summarization. *Inf. Process. Manage.*, 47(2):227–237, March.
- Prasad Pingali, Rahul K, and Vasudeva Varma, 2007. *IIIT Hyderabad at DUC 2007*. U.S. National Inst. of Standards and Technology.
- Chao Shen and Tao Li. 2011. Learning to rank for query-focused multi-document summarization. In Diane J. Cook, Jian Pei, Wei Wang 0010, Osmar R. Zaane, and Xindong Wu, editors, *ICDM*, pages 626–634. IEEE.
- Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 2, pages 901–904, Denver, USA.
- Kristina Toutanova, Chris Brockett, Michael Gamon, Jagadeesh Jagarlamudi, Hisami Suzuki, and Lucy Vanderwende. 2007. The PYTHY Summarization System: Microsoft Research at DUC 2007. In *Proc. of DUC*.
- Jenine Turner and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. *ACL '05*, pages 290–297, Stroudsburg, PA, USA. Association for Computational Linguistics.
- David Zajic, Bonnie J Dorr, Jimmy Lin, and R. Schwartz. 2006. Sentence compression as a component of a multi-document summarization system. *Proceedings of the 2006 Document Understanding Workshop, New York*.

# Domain-Independent Abstract Generation for Focused Meeting Summarization

Lu Wang

Department of Computer Science  
Cornell University  
Ithaca, NY 14853  
luwang@cs.cornell.edu

Claire Cardie

Department of Computer Science  
Cornell University  
Ithaca, NY 14853  
cardie@cs.cornell.edu

## Abstract

We address the challenge of generating natural language abstractive summaries for spoken meetings in a domain-independent fashion. We apply *Multiple-Sequence Alignment* to induce abstract generation templates that can be used for different domains. An *Overgenerate-and-Rank* strategy is utilized to produce and rank candidate abstracts. Experiments using in-domain and out-of-domain training on disparate corpora show that our system uniformly outperforms state-of-the-art supervised extract-based approaches. In addition, human judges rate our system summaries significantly higher than compared systems in fluency and overall quality.

## 1 Introduction

Meetings are a common way to collaborate, share information and exchange opinions. Consequently, automatically generated meeting summaries could be of great value to people and businesses alike by providing quick access to the essential content of past meetings. *Focused meeting summaries* have been proposed as particularly useful; in contrast to summaries of a meeting as a whole, they refer to summaries of a specific aspect of a meeting, such as the DECISIONS reached, PROBLEMS discussed, PROGRESS made or ACTION ITEMS that emerged (Carenini et al., 2011). Our goal is to provide an automatic summarization system that can generate abstract-style focused meeting summaries to help users digest the vast amount of meeting content in an easy manner.

Existing meeting summarization systems remain largely *extractive*: their summaries are comprised exclusively of patchworks of utterances selected directly from the meetings to be summarized (Riedhammer et al., 2010; Bui et al., 2009; Xie et al., 2008). Although relatively easy to construct, extractive approaches fall short of producing concise and readable summaries, largely due

C: Looking at what we've got, we we want an LCD display with a spinning wheel.  
B: You have to have some push-buttons, don't you?  
C: Just spinning and not scrolling, I would say.  
B: I think the spinning wheel is definitely very now.  
A: but since LCDs seems to be uh a definite yes,  
C: We're having push-buttons on the outside  
C: and then on the inside an LCD with spinning wheel,

### Decision Abstract (Summary):

The remote will have push buttons outside, and an LCD and spinning wheel inside.

A: and um I'm not sure about the buttons being in the shape of fruit though.

D: Maybe make it like fruity colours or something.

C: The power button could be like a big apple or something.

D: Um like I'm just thinking bright colours.

### Problem Abstract (Summary):

How to incorporate a fruit and vegetable theme into the remote.

Figure 1: Clips from the AMI meeting corpus (McCowan et al., 2005). A, B, C and D refer to distinct speakers. Also shown is the gold-standard (manual) abstract (summary) for the decision and the problem.

to the noisy, fragmented, ungrammatical and unstructured text of meeting transcripts (Murray et al., 2010b; Liu and Liu, 2009).

In contrast, human-written meeting summaries are typically in the form of *abstracts* — distillations of the original conversation written in new language. A user study from Murray et al. (2010b) showed that people demonstrate a strong preference for abstractive summaries over extracts when the text to be summarized is conversational. Consider, for example, the two types of focused summary along with their associated dialogue snippets in Figure 1. We can see that extracts are likely to include unnecessary and noisy information from the meeting transcripts. On the contrary, the manually composed summaries (abstracts) are more compact and readable, and are written in a distinctly non-conversational style.

To address the limitations of extract-based summaries, we propose a complete and fully automatic domain-independent abstract generation framework for focused meeting summarization. Following existing language generation research (Angeli et al., 2010; Konstas and Lapata, 2012), we first perform *content selection*: given the dialogue acts relevant to one element of the meeting (e.g. a single decision or problem), we train a classifier to identify summary-worthy phrases. Next, we develop an “overgenerate-and-rank” strategy (Walker et al., 2001; Heilman and Smith, 2010) for *surface realization*, which generates and ranks candidate sentences for the abstract. After redundancy reduction, the full meeting abstract can thus comprise the focused summary for each meeting element. As described in subsequent sections, the generation framework allows us to identify and reformulate the important information for the focused summary. Our contributions are as follows:

- To the best of our knowledge, our system is the first fully automatic system to generate natural language abstracts for spoken meetings.
- We present a novel template extraction algorithm, based on Multiple Sequence Alignment (MSA) (Durbin et al., 1998), to induce domain-independent templates that guide abstract generation. MSA is commonly used in bioinformatics to identify equivalent fragments of DNAs (Durbin et al., 1998) and has also been employed for learning paraphrases (Barzilay and Lee, 2003).
- Although our framework requires labeled training data for each type of focused summary (decisions, problems, etc.), we also make initial tries for domain adaptation so that our summarization method does not need human-written abstracts for each new meeting domain (e.g. faculty meetings, theater group meetings, project group meetings).

We instantiate the abstract generation framework on two corpora from disparate domains — the AMI Meeting Corpus (Mccowan et al., 2005) and ICSI Meeting Corpus (Janin et al., 2003) — and produce systems to generate focused summaries with regard to four types of

meeting elements: DECISIONS, PROBLEMS, ACTION ITEMS, and PROGRESS. Automatic evaluation (using ROUGE (Lin and Hovy, 2003) and BLEU (Papineni et al., 2002)) against manually generated focused summaries shows that our summarizers uniformly and statistically significantly outperform two baseline systems as well as a state-of-the-art supervised extraction-based system. Human evaluation also indicates that the abstractive summaries produced by our systems are more linguistically appealing than those of the utterance-level extraction-based system, preferring them over summaries from the extraction-based system of comparable semantic correctness (62.3% vs. 37.7%).

Finally, we examine the generality of our model across domains for two types of focused summarization — decisions and problems — by training the summarizer on out-of-domain data (i.e. the AMI corpus for use on the ICSI meeting data, and vice versa). The resulting systems yield results comparable to those from the same system trained on in-domain data, and statistically significantly outperform supervised extractive summarization approaches trained on in-domain data.

## 2 Related Work

Most research on spoken dialogue summarization attempts to generate summaries for full dialogues (Carenini et al., 2011). Only recently has the task of focused summarization been studied. Supervised methods are investigated to identify key phrases or utterances for inclusion in the decision summary (Fernández et al., 2008; Bui et al., 2009). Based on Fernández et al. (2008), a relation representation is proposed by Wang and Cardie (2012) to form structured summaries; we adopt this representation here for content selection.

Our research is also in line with generating abstractive summaries for conversations. Extractive approaches (Murray et al., 2005; Xie et al., 2008; Galley, 2006) have been investigated extensively in conversation summarization. Murray et al. (2010a) present an abstraction system consisting of interpretation and transformation steps. Utterances are mapped to a simple conversation ontology in the interpretation step according to their type, such as a decision or problem. Then an integer linear programming approach is employed to select the utterances that cover more entities as

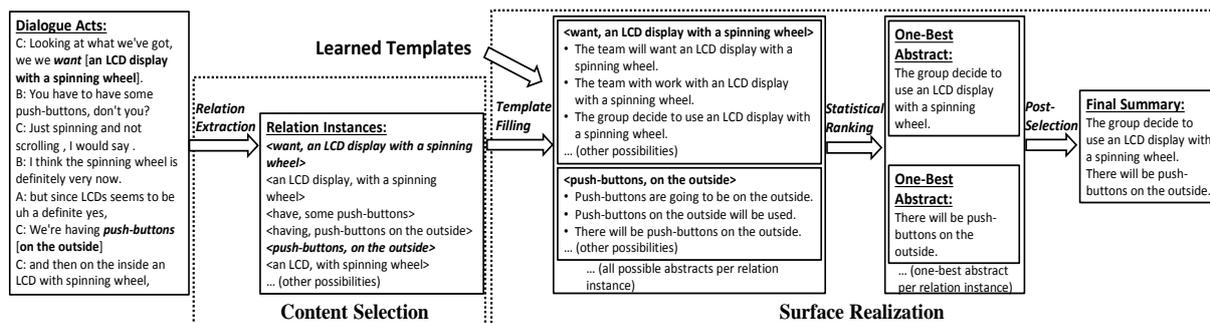


Figure 2: The abstract generation framework. It takes as input a cluster of meeting-item-specific dialogue acts, from which one focused summary is constructed. Sample relation instances are denoted in **bold** (The indicators are further *italicized* and the arguments are in [brackets]). Summary-worthy relation instances are identified by content selection module (see Section 4) and then filled into the learned templates individually. A statistical ranker subsequently selects one best abstract per relation instance (see Section 5.2). The post-selection component reduces the redundancy and outputs the final summary (see Section 5.3).

determined by an external ontology. Liu and Liu (2009) apply sentence compression on extracted summary utterances. Though some of the unnecessary words are dropped, the resulting compressions can still be ungrammatical and unstructured.

This work is also broadly related to expert system-based language generation (Reiter and Dale, 2000) and concept-to-text generation tasks (Angeli et al., 2010; Konstas and Lapata, 2012), where the generation process is decomposed into content selection (or text planning) and surface realization. For instance, Angeli et al. (2010) learn from structured database records and parallel textual descriptions. They generate texts based on a series of decisions made to select the records, fields, and proper templates for rendering. Those techniques that are tailored to specific domains (e.g. weather forecasts or sportcastings) cannot be directly applied to the conversational data, as their input is well-structured and the templates learned are domain-specific.

### 3 Framework

Our domain-independent abstract generation framework produces a summarizer that generates a grammatical abstract from a cluster of *meeting-element-related dialogue acts (DAs)* — all utterances associated with a single decision, problem, action item or progress step of interest. Note that identifying these DA clusters is a difficult task in itself (Bui et al., 2009). Accordingly, our experiments evaluate two conditions — one in which we assume that they are perfectly identified, and one in which we identify the clusters automatically.

The summarizer consists of two major components and is depicted in Figure 2. Given the DA cluster to be summarized, the *Content Selection* module identifies a set of summary-worthy *relation instances* represented as indicator-argument pairs (i.e. these constitute a finer-grained representation than DAs). The *Surface Realization* component then generates a short summary in three steps. In the first step, each relation instance is filled into templates with disparate structures that are learned automatically from the training set (*Template Filling*). A statistical ranker then selects one best abstract per relation instance (*Statistical Ranking*). Finally, selected abstracts are processed for redundancy removal in *Post-Selection*. Detailed descriptions for each individual step are provided in Sections 4 and 5.

### 4 Content Selection

Phrase-based content selection approaches have been shown to support better meeting summaries (Fernández et al., 2008). Therefore, we chose a content selection representation of a finer granularity than an utterance: we identify *relation instances* that can both effectively detect the crucial content and incorporate enough syntactic information to facilitate the downstream surface realization.

More specifically, our relation instances are based on information extraction methods that identify a lexical *indicator* (or *trigger*) that evokes a relation of interest and then employ syntactic information, often in conjunction with semantic constraints, to find the *argument constituent* (or *target phrase*) to be extracted. Rela-

tion instances, then, are represented by **indicator-argument** pairs (Chen et al., 2011). For example, in the DA cluster of Figure 2, *⟨want, an LCD display with a spinning wheel⟩* and *⟨push-buttons, on the outside⟩* are two relation instances.

**Relation Instance Extraction** We adopt and extend the syntactic constraints from Wang and Cardie (2012) to identify all relation instances in the input utterances; the summary-worthy ones will be selected by a discriminative classifier. Constituent and dependency parses are obtained by the Stanford parser (Klein and Manning, 2003). Both the indicator and argument take the form of constituents in the parse tree. We restrict the eligible indicator to be a noun or verb; the eligible arguments is a noun phrase (NP), prepositional phrase (PP) or adjectival phrase (ADJP). A valid indicator-argument pair should have at least one content word and satisfy one of the following constraints:

- When the indicator is a noun, the argument has to be a modifier or complement of the indicator.
- When the indicator is a verb, the argument has to be the subject or the object if it is an NP, or a modifier or complement of the indicator if it is a PP/ADJP.

We view relation extraction as a binary classification problem rather than a clustering task (Chen et al., 2011). All relation instances can be categorized as summary-worthy or not, but only the summary-worthy ones are used for abstract generation. A discriminative classifier is trained for this purpose based on Support Vector Machines (SVMs) (Joachims, 1998) with an RBF kernel. For training data construction, we consider a relation instance to be a positive example if it shares any content word with its corresponding abstracts, and a negative example otherwise. The features used are shown in Table 1.

## 5 Surface Realization

In this section, we describe surface realization, which renders the relation instances into natural language abstracts. This process begins with template extraction (Section 5.1). Once the templates are learned, the relation instances from Section 4 are filled into the templates to generate an abstract (see Section 5.2). Redundancy handling is discussed in Section 5.3.

<b>Basic Features</b>
number of words/content words
portion of content words/stopwords
number of content words in indicator/argument
number of content words that are also in previous DA
indicator/argument only contains stopword?
number of new nouns
<b>Content Features</b>
has capitalized word?
has proper noun?
TF/IDF/TFIDF min/max/average
<b>Discourse Features</b>
main speaker or not?
is in an adjacency pair (AP)?
is in the source/target of the AP?
number of source/target DA in the AP
is the target of the AP a positive/negative/neutral response?
is the source of the AP a question?
<b>Syntax Features</b>
indicator/argument constituent tag
dependency relation of indicator and argument

Table 1: Features for content selection. Most are adapted from previous work (Galley, 2006; Xie et al., 2008; Wang and Cardie, 2012). Every basic or content feature is concatenated with the constituent tags of indicator and argument to compose a new one. Main speakers include the most talkative speaker (who has said the most words) and other speakers whose word count is more than 20% of the most talkative one (Xie et al., 2008). Adjacency pair (AP) (Galley, 2006) is an important conversational analysis concept; each AP consists of a source utterance and a target utterance produced by different speakers.

### 5.1 Template Extraction

**Sentence Clustering.** Template extraction starts with clustering the sentences that constitute the manually generated abstracts in the training data according to their lexical and structural similarity. From each cluster, multiple-sequence alignment techniques are employed to capture the recurring patterns.

Intuitively, desirable templates are those that can be applied in different domains to generate the same type of focused summary (e.g. decision or problem summaries). We do not want sentences to be clustered only because they describe the same domain-specific details (e.g. they are all about “data collection”), which will lead to fragmented templates that are not reusable for new domains. We therefore replace all appearances of dates, numbers, and proper names with generic labels. We also replace words that appear in both the abstract and supporting dialogue acts by a label indicating its phrase type. For any noun phrase with its head word abstracted, the whole phrase is also replaced with “NP”.

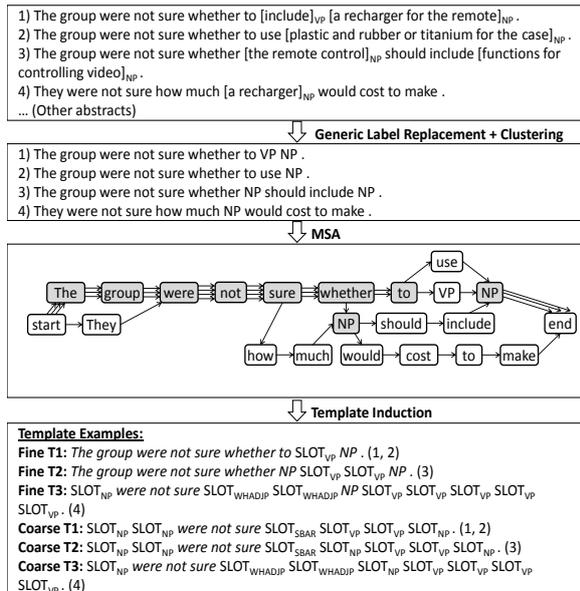


Figure 3: Example of template extraction by Multiple-Sequence Alignment for problem abstracts from AMI. Backbone nodes shared by at least 50% sentences are shaded. The grammatical errors exist in the original abstracts.

Following Barzilay and Lee (2003), we approach the sentence clustering task by hierarchical complete-link clustering with a similarity metric based on word n-gram overlap ( $n = 1, 2, 3$ ). Clusters with fewer than three abstracts are removed<sup>1</sup>.

**Learning the Templates via MSA.** For learning the structural patterns among the abstracts, *Multiple-Sequence Alignment (MSA)* is first computed for each cluster. MSA takes as input multiple sentences and one scoring function to measure the similarity between any two words. For insertions or deletions, a gap cost is also added. MSA can thus find the best way to align the sequences with insertions or deletions in accordance with the scorer. However, computing an optimal MSA is NP-complete (Wang and Jiang, 1994), thus we implement an approximate algorithm (Needleman and Wunsch, 1970) that iteratively aligns two sequences each time and treats the resulting alignment as a new sequence<sup>2</sup>. Figure 3 demonstrates an MSA computed from a sample cluster of ab-

<sup>1</sup>Clustering stops when the similarity between any pairwise clusters is below 5. This is applied to every type of summarization. We tune the parameter on a small held-out development set by manually evaluating the induced templates. No significant change is observed within a small range.

<sup>2</sup>We adopt the scoring function for MSA from Barzilay and Lee (2003), where aligning two identical words scores 1, inserting a gap scores  $-0.01$ , and aligning two different words scores  $-0.5$ .

stracts. The MSA is represented in the form of word lattice, from which we can detect the structural similarities shared by the sentences.

To transform the resulting MSAs into templates, we need to decide whether a word in the sentence should be retained to comprise the template or abstracted. The *backbone* nodes in an MSA are identified as the ones shared by more than 50%<sup>3</sup> of the cluster’s sentences (shaded in gray in Figure 3). We then create a FINE template for each sentence by abstracting the non-backbone words, i.e. replacing each of those words with a generic token (last step in Figure 3). We also create a COARSE template that only preserves the nodes shared by all of the cluster’s sentences. By using the operations above, domain-independent patterns are thus identified and domain-specific details are removed.

Note that we do not explicitly evaluate the quality of the learned templates, which would require a significant amount of manual evaluation. Instead, they are evaluated extrinsically. We encode the templates as features (Angeli et al., 2010) that could be selected or ignored in the succeeding abstract ranking model.

## 5.2 Template Filling

**An Overgenerate-and-Rank Approach.** Since filling the relation instances into templates of distinct structures may result in abstracts of varying quality, we rank the abstracts based on the features of the template, the transformation conducted, and the generated abstract. This is realized by the *Overgenerate-and-Rank* strategy (Walker et al., 2001; Heilman and Smith, 2010). It takes as input a set of relation instances (from the same cluster)  $R = \{\langle ind_i, arg_i \rangle\}_{i=1}^N$  that are produced by content selection component, a set of templates  $T = \{t_j\}_{j=1}^M$  that are represented as parsing trees, a transformation function  $F$  (described below), and a statistical ranker  $S$  for ranking the generated abstracts, for which we defer description later in this Section.

For each  $\langle ind_i, arg_i \rangle$ , the overgenerate-and-rank approach fills it into each template in  $T$  by applying  $F$  to generate all possible abstracts. Then the ranker  $S$  selects the best abstract  $abs_i$ . Post-selection is conducted on the abstracts  $\{abs_i\}_{i=1}^N$  to form the final summary.

<sup>3</sup>See Barzilay and Lee (2003) for a detailed discussion about the choice of 50% according to pigeonhole principle.

The transformation function  $F$  models the *constituent-level* transformations of relation instances and their mappings to the parse trees of templates. With the intuition that people will reuse the relation instances from the transcripts albeit not necessarily in their original form to write the abstracts, we consider three major types of mapping operations for the indicator or argument in the source pair, namely, *Full-Constituent Mapping*, *Sub-Constituent Mapping*, and *Removal*. *Full-Constituent Mapping* denotes that a source constituent is mapped directly to a target constituent of the template parse tree with the same tag. *Sub-Constituent Mapping* encodes more complex and flexible transformations in that a sub-constituent of the source is mapped to a target constituent with the same tag. This operation applies when the source has a tag of PP or ADJP, in which case its sub-constituent, if any, with a tag of NP, VP or ADJP can be mapped to the target constituent with the same tag. For instance, an argument “with a spinning wheel” (PP) can be mapped to an NP in a template because it has a sub-constituent “a spinning wheel” (NP). *Removal* means a source is not mapped to any constituent in the template.

Formally,  $F$  is defined as:

$$F(\langle ind^{src}, arg^{src} \rangle, t) = \{\langle ind_k^{tran}, arg_k^{tran}, ind_k^{tar}, arg_k^{tar} \rangle\}_{k=1}^K$$

where  $\langle ind^{src}, arg^{src} \rangle \in R$  is a relation instance (*source pair*);  $t \in T$  is a template;  $ind_k^{tran}$  and  $arg_k^{tran}$  is the *transformed pair* of  $ind^{src}$  and  $arg^{src}$ ;  $ind_k^{tar}$  and  $arg_k^{tar}$  are constituents in  $t$ , and they compose one *target pair* for  $\langle ind^{src}, arg^{src} \rangle$ . We require that  $ind^{src}$  and  $arg^{src}$  are not removed at the same time. Moreover, for valid  $ind_k^{tar}$  and  $arg_k^{tar}$ , the words subsumed by them should be all abstracted in the template, and they do not overlap in the parse tree.

To obtain the realized abstract, we traverse the parse tree of the filled template in pre-order. The words subsumed by the leaf nodes are thus collected sequentially.

**Learning a Statistical Ranker.** We utilize a discriminative ranker based on Support Vector Regression (SVR) (Smola and Schölkopf, 2004) to rank the generated abstracts. Given the training data that includes clusters of gold-standard summary-worthy relation instances, associated abstracts they support, and the parallel templates for each abstract, training samples for the ranker are

<p><b>Basic Features</b>            number of words in <math>ind^{src}/arg^{src}</math>            number of new nouns in <math>ind^{src}/arg^{src}</math>  <math>ind_k^{tran}/arg_k^{tran}</math> only has stopword?            number of new nouns in <math>ind_k^{tran}/arg_k^{tran}</math></p>
<p><b>Structure Features</b>            constituent tag of <math>ind^{src}/arg^{src}</math>            constituent tag of <math>ind^{src}</math> with constituent tag of <math>ind^{tar}</math>            constituent tag of <math>arg^{src}</math> with constituent tag of <math>arg^{tar}</math>            transformation of <math>ind^{src}/arg^{src}</math> combined with constituent tag            dependency relation of <math>ind^{src}</math> and <math>arg^{src}</math>            dependency relation of <math>ind^{tar}</math> and <math>arg^{tar}</math>            above 2 features have same value?</p>
<p><b>Template Features</b>            template type (fine/coarse)            realized template (e.g. “the group decided to”)            number of words in template            the template has verb?</p>
<p><b>Realization Features</b>            realization has verb?            realization starts with verb?            realization has adjacent verbs/NPs?  <math>ind^{src}</math> precedes/succeeds <math>arg^{src}</math>?  <math>ind^{tar}</math> precedes/succeeds <math>arg^{tar}</math>?            above 2 features have same value?</p>
<p><b>Language Model Features</b>  <math>\log p_{LM}</math>(first word in <math>ind_k^{tran}</math>   previous 1/2 words)  <math>\log p_{LM}</math>(realization)  <math>\log p_{LM}</math>(first word in <math>arg_k^{tran}</math>   previous 1/2 words)  <math>\log p_{LM}</math>(realization)/length  <math>\log p_{LM}</math>(next word   last 1/2 words in <math>ind_k^{tran}</math>)  <math>\log p_{LM}</math>(next word   last 1/2 words in <math>arg_k^{tran}</math>)</p>

Table 2: Features for abstracts ranking. The language model features are based on a 5-gram language model trained on Gigaword (Graff, 2003) by SRILM (Stolcke, 2002).

constructed according to the transformation function  $F$  mentioned above. Each sample is represented as:

$$(\langle ind^{src}, arg^{src} \rangle, \langle ind_k^{tran}, arg_k^{tran}, ind_k^{tar}, arg_k^{tar} \rangle, t, a)$$

where  $\langle ind^{src}, arg^{src} \rangle$  is the source pair,  $\langle ind_k^{tran}, arg_k^{tran} \rangle$  is the transformed pair,  $\langle ind_k^{tar}, arg_k^{tar} \rangle$  is the target pair in template  $t$ , and  $a$  is the abstract parallel to  $t$ .

We first find  $\langle ind_k^{tar,abs}, arg_k^{tar,abs} \rangle$ , which is the corresponding constituent pair of  $\langle ind_k^{tar}, arg_k^{tar} \rangle$  in  $a$ . Then we identify the summary-worthy words subsumed by  $\langle ind_k^{tran}, arg_k^{tran} \rangle$  that also appear in  $a$ . If those words are all subsumed by  $\langle ind_k^{tar,abs}, arg_k^{tar,abs} \rangle$ , then it is considered to be a positive sample, and a negative sample otherwise. Table 2 displays the features used in abstract ranking.

### 5.3 Post-Selection: Redundancy Handling.

Post-selection aims to maximize the information coverage and minimize the redundancy of the summary. Given the generated abstracts  $A =$

**Input** : relation instances  $R = \{\langle ind_i, arg_i \rangle\}_{i=1}^N$ ,  
generated abstracts  $A = \{abs_i\}_{i=1}^N$ , objective  
function  $f$ , cost function  $C$

**Output**: final abstract  $G$

$G \leftarrow \Phi$  (empty set);  
 $U \leftarrow A$ ;  
**while**  $U \neq \Phi$  **do**  
     $abs \leftarrow \arg \max_{abs_i \in U} \frac{f(A, G \cup abs_i) - f(A, G)}{C(abs_i)}$ ;  
    **if**  $f(A, G \cup abs) - f(A, G) \geq 0$  **then**  
         $G \leftarrow G \cup abs$ ;  
    **end**  
     $U \leftarrow U \setminus abs$ ;  
**end**

**Algorithm 1:** Greedy algorithm for post-selection to generate the final summary.

$\{abs_i\}_{i=1}^N$ , we use a greedy algorithm (Lin and Bilmes, 2010) to select a subset  $A'$ , where  $A' \subseteq A$ , to form the final summary. We define  $w_{ij}$  as the unigram similarity between abstracts  $abs_i$  and  $abs_j$ ,  $C(abs_i)$  as the number of words in  $abs_i$ . We employ the following objective function:

$$f(A, G) = \sum_{abs_i \in A \setminus G} \sum_{abs_j \in G} w_{i,j}, G \subseteq A$$

Algorithm 1 sequentially finds an abstract with the greatest ratio of objective function gain to length, and add it to the summary if the gain is non-negative.

## 6 Experimental Setup

**Corpora.** Two disparate corpora are used for evaluation. The AMI meeting corpus (Mccowan et al., 2005) contains 139 scenario-driven meetings, where groups of four people participate in a series of four meetings for a fictitious project of designing remote control. The ICSI meeting corpus (Janin et al., 2003) consists of 75 naturally occurring meetings, each of them has 4 to 10 participants. Compared to the fabricated topics in AMI, the conversations in ICSI tend to be specialized and technical, e.g. discussion about speech and language technology. We use 57 meetings in ICSI and 139 meetings in AMI that include a short (usually one-sentence), manually constructed abstract summarizing each important output for every meeting. Decision and problem summaries are annotated for both corpora. AMI has extra action item summaries, and ICSI has progress summaries. The set of dialogue acts that support each abstract are annotated as such.

**System Inputs.** We consider two system input settings. In the **True Clusterings** setting, we use the annotations to create perfect partitions of the DAs for input to the system; in the **System**

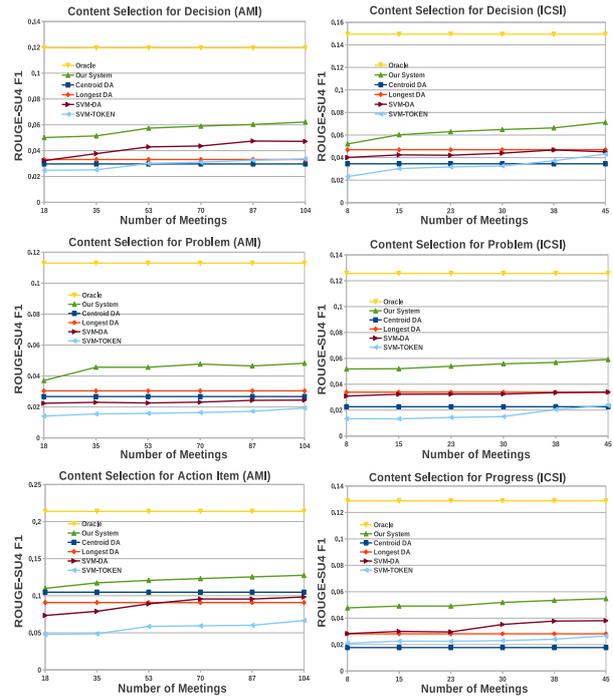


Figure 4: Content selection evaluation by using ROUGE-SU4 (multiplied by 100). SVM-DA and SVM-TOKEN denotes for supervised extract-based methods with SVMs on utterance- and token-level. Summaries for decision, problem, action item, and progress are generated and evaluated for AMI and ICSI (with names in parentheses). X-axis shows the number of meetings used for training.

**Clusterings** setting, we employ a hierarchical agglomerative clustering algorithm used for this task in (Wang and Cardie, 2011). DAs are grouped according to a classifier trained beforehand.

**Baselines and Comparisons.** We compare our system with (1) two unsupervised baselines, (2) two supervised extractive approaches, and (3) an oracle derived from the gold standard abstracts.

**Baselines.** As in Riedhammer et al. (2010), the LONGEST DA in each cluster is selected as the summary. The second baseline picks the cluster prototype (i.e. the DA with the largest TF-IDF similarity with the cluster centroid) as the summary according to Wang and Cardie (2011). Although it is possible that important content is spread over multiple DAs, both baselines allow us to determine summary quality when summaries are restricted to a single utterance.

**Supervised Learning.** We also compare our approach to two supervised extractive summarization methods — Support Vector Machines (Joachims, 1998) trained with the same fea-

tures as our system (see Table 1) to identify the important **DAs** (no syntax features) (Xie et al., 2008; Sandu et al., 2010) or **tokens** (Fernández et al., 2008) to include into the summary<sup>4</sup>.

*Oracle.* We compute an oracle consisting of the words from the DA cluster that also appear in the associated abstract to reflect the gap between the best possible extracts and the human abstracts.

## 7 Results

**Content Selection Evaluation.** We first employ ROUGE (Lin and Hovy, 2003) to evaluate the content selection component with respect to the human written abstracts. ROUGE computes the ngram overlapping between the system summaries with the reference summaries, and has been used for both text and speech summarization (Dang, 2005; Xie et al., 2008). We report ROUGE-2 (R-2) and ROUGE-SU4 (R-SU4) that are shown to correlate with human evaluation reasonably well.

In AMI, four meetings of different functions are carried out in each group<sup>5</sup>. 35 meetings for “conceptual design” are randomly selected for testing. For ICSI, we reserve 12 meetings for testing.

The R-SU4 scores for each system are displayed in Figure 4 and show that our system uniformly outperforms the baselines and supervised systems. The learning curve of our system is relatively flat, which means not many training meetings are required to reach a usable performance level.

Note that the ROUGE scores are relative low when the reference summaries are human abstracts, even for evaluation among abstracts produced by different annotators (Dang, 2005). The intrinsic difference of styles between dialogue and human abstract further lowers the scores. But the trend is still respected among the systems.

**Abstract Generation Evaluation.** To evaluate the full abstract generation system, the BLEU score (Papineni et al., 2002) (the precision of unigrams and bigrams with a brevity penalty) is computed with human abstracts as reference. BLEU has a fairly good agreement with human judgement and has been used to evaluate a variety of language generation systems (Angeli et al., 2010; Konstas and Lapata, 2012).

<sup>4</sup>We use  $SVM^{light}$  (Joachims, 1999) with RBF kernel by default parameters for SVM-based classifiers and regressor.

<sup>5</sup>The four types of meetings in AMI are: project kick-off (35 meetings), functional design (35 meetings), conceptual design (35 meetings), and detailed design (34 meetings).

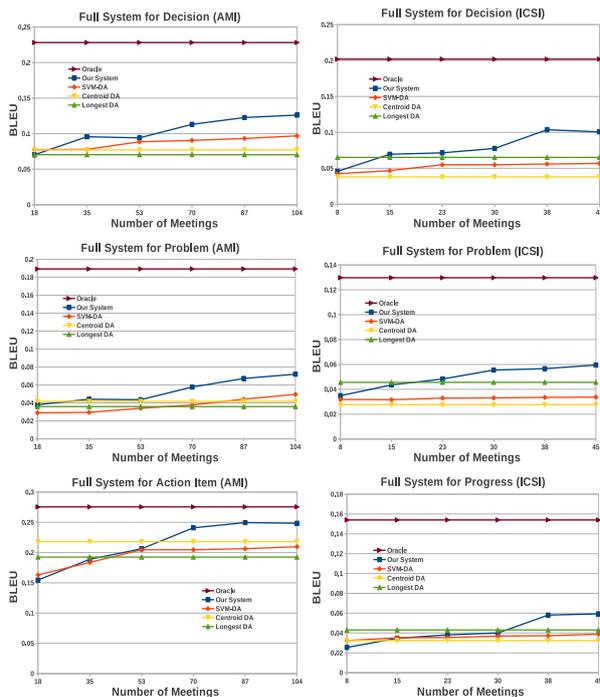


Figure 5: Full abstract generation system evaluation by using BLEU (multiplied by 100). SVM-DA denotes for supervised extractive methods with SVMs on utterance-level.

We are not aware of any existing work generating abstractive summaries for conversations. Therefore, we compare our full system against a supervised utterance-level extractive method based on SVMs along with the baselines. The BLEU scores in Figure 5 show that our system improves the scores consistently over the baselines and the SVM-based approach.

**Domain Adaptation Evaluation.** We further examine our system in domain adaptation scenarios for decision and problem summarization, where we train the system on AMI for use on ICSI, and vice versa. Table 3 indicates that, with both true clusterings and system clusterings, our system trained on out-of-domain data achieves comparable performance with the same system trained on in-domain data. In most experiments, it also significantly outperforms the baselines and the extract-based approaches ( $p < 0.05$ ).

**Human Evaluation.** We randomly select 15 decision and 15 problem DA clusters (true clusterings). We evaluate **fluency** (is the text grammatical?) and **semantic correctness** (does the summary convey the gist of the DAs in the cluster?) for OUR SYSTEM trained on IN-domain data

System (True Clusterings)	AMI Decision			ICSI Decision			AMI Problem			ICSI Problem		
	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>
CENTROID DA	1.3	3.0	7.7	1.8	3.5	3.8	1.0	2.7	4.2	1.0	2.3	2.8
LONGEST DA	1.6	3.3	7.0	2.8	4.7	6.5	1.0	3.0	3.6	1.2	3.4	4.6
SVM-DA (IN)	3.4	4.7	9.7	3.4	4.5	5.7	1.4	2.4	5.0	1.6	3.4	3.4
SVM-DA (OUT)	2.7	4.2	6.6	3.1	4.2	4.6	1.4	2.2	2.5	1.3	3.0	4.6
OUR SYSTEM (IN)	<b>4.5</b>	<b>6.2</b>	<b>11.6</b>	<b>4.9</b>	<b>7.1</b>	<b>10.0</b>	<b>3.1</b>	<b>4.8</b>	<b>7.2</b>	<b>4.0</b>	<b>5.9</b>	<b>6.0</b>
OUR SYSTEM (OUT)	<b>4.6</b>	<b>6.1</b>	<b>10.3</b>	<b>4.8</b>	<b>6.4</b>	<b>7.8</b>	<b>3.5</b>	<b>4.7</b>	<b>6.2</b>	<b>3.0</b>	<b>5.5</b>	<b>5.3</b>
ORACLE	7.5	12.0	22.8	9.9	14.9	20.2	6.6	11.3	18.9	6.4	12.6	13.0
System (System Clusterings)	AMI Decision			ICSI Decision			AMI Problem			ICSI Problem		
	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>	<i>R-2</i>	<i>R-SU4</i>	<i>BLEU</i>
CENTROID DA	1.4	3.3	3.8	1.4	2.1	2.0	0.8	2.8	2.9	0.9	2.3	1.8
LONGEST DA	1.4	3.3	5.7	1.7	3.4	5.5	0.8	3.2	4.1	0.9	3.4	4.4
SVM-DA (IN)	2.6	4.6	10.5	3.5	6.5	7.1	1.8	3.7	4.9	1.8	4.0	4.6
SVM-DA (OUT)	3.4	5.8	10.3	2.7	4.8	6.3	2.1	3.8	4.3	1.5	3.8	3.5
OUR SYSTEM (IN)	<b>3.5</b>	<b>5.4</b>	<b>11.7</b>	<b>4.4</b>	<b>7.4</b>	<b>9.1</b>	<b>3.3</b>	<b>4.6</b>	<b>9.5</b>	<b>2.3</b>	<b>4.2</b>	<b>7.4</b>
OUR SYSTEM (OUT)	<b>3.9</b>	<b>6.4</b>	<b>11.4</b>	<b>4.1</b>	<b>5.1</b>	<b>8.4</b>	<b>3.6</b>	<b>5.6</b>	<b>8.9</b>	<b>1.8</b>	<b>4.0</b>	<b>6.8</b>
ORACLE	6.4	12.0	15.1	8.2	15.2	17.6	6.5	13.0	20.9	5.5	11.9	15.5

Table 3: Domain adaptation evaluation. Systems trained on out-of-domain data are denoted with “(OUT)”, otherwise with “(IN)”. ROUGE and BLEU scores are multiplied by 100. Our systems that statistically significantly outperform all the other approaches (except ORACLE) are in **bold** ( $p < 0.05$ , paired  $t$ -test). The numbers in *italics* show the significant improvement over the baselines by our systems.

System	Fluency		Semantic		Length
	<i>Mean</i>	<i>S.D.</i>	<i>Mean</i>	<i>S.D.</i>	
OUR SYSTEM (IN)	<b>3.67</b>	0.85	3.27	1.03	<b>23.65</b>
OUR SYSTEM (OUT)	3.58	0.90	3.25	1.16	<b>24.17</b>
SVM-DA (IN)	3.36	0.84	3.44	1.26	38.83

Table 4: Human evaluation results of **Fluency** and **Semantic** correctness for the generated abstracts. The ratings are on 1 (worst) to 5 (best) scale. The average **Length** of the abstracts for each system is also listed.

and OUT-of-domain data, and for the utterance-level extraction system (SVM-DA) trained on in-domain data. Each cluster of DAs along with three randomly ordered summaries are presented to the judges. Five native speaking Ph.D. students (none are authors) performed the task.

We carry out an one-way Analysis of Variance which shows significant differences in score as a function of system ( $p < 0.05$ , paired  $t$ -test). Results in Table 4 demonstrate that our system summaries are significantly more compact and fluent than the extract-based method ( $p < 0.05$ ) while semantic correctness is comparable.

The judges also **rank** the three summaries in terms of the overall quality in content, conciseness and grammaticality. An inter-rater agreement of Fleiss’s  $\kappa = 0.45$  (moderate agreement (Landis and Koch, 1977)) was computed. Judges selected our system as the best system in 62.3% scenarios (IN-DOMAIN: 35.6%, OUT-OF-DOMAIN: 26.7%). Sample summaries are exhibited in Figure 6.

## 8 Conclusion

We presented a domain-independent abstract generation framework for focused meeting summarization. Experimental results on two disparate meeting corpora show that our system can uni-

Decision Summary:
<b>Human:</b> The remote will have push buttons outside, and an LCD and spinning wheel inside.
<b>Our System (In):</b> The group decide to use an LCD display with a spinning wheel. There will be push-buttons on the outside.
<b>Our System (Out):</b> LCD display is going to be with a spinning wheel. It is necessary having push-buttons on the outside.
<b>SVM-DA:</b> Looking at what we’ve got, we we want an LCD display with a spinning wheel. Just spinning and not scrolling, I would say. I think the spinning wheel is definitely very now. We’re having push-buttons on the outside
Problem Summary:
<b>Human:</b> How to incorporate a fruit and vegetable theme into the remote.
<b>Our System (In):</b> Whether to include the shape of fruit. The team had to thinking bright colors.
<b>Our System (Out):</b> It is unclear that the buttons being in the shape of fruit.
<b>SVM-DA:</b> and um Im not sure about the buttons being in the shape of fruit though.

Figure 6: Sample decision and problem summaries generated by various systems for examples in Figure 1.

formly outperform the state-of-the-art supervised extraction-based systems in both automatic and manual evaluation. Our system also exhibits an ability to train on out-of-domain data to generate abstracts for a new target domain.

## 9 Acknowledgments

This work was supported in part by National Science Foundation Grant IIS-0968450 and a gift from Boeing. We thank Moontae Lee, Myle Ott, Yiye Ruan, Chenhao Tan, and the ACL reviewers for valuable suggestions and advice on various aspects of this work.

## References

- Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pages 502–512, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1, NAACL '03*, pages 16–23, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Trung H. Bui, Matthew Frampton, John Dowding, and Stanley Peters. 2009. Extracting decisions from multi-party dialogue using directed graphical models and semantic similarity. In *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL '09*, pages 235–243, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Giuseppe Carenini, Gabriel Murray, and Raymond Ng. 2011. *Methods for Mining and Summarizing Text Conversations*. Morgan & Claypool Publishers.
- Harr Chen, Edward Benson, Tahira Naseem, and Regina Barzilay. 2011. In-domain relation discovery with meta-constraints via posterior regularization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11*, pages 530–540, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hoa T. Dang. 2005. Overview of DUC 2005. In *Document Understanding Conference*.
- Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, July.
- Raquel Fernández, Matthew Frampton, John Dowding, Anish Adukuzhiyil, Patrick Ehlen, and Stanley Peters. 2008. Identifying relevant phrases to summarize decisions in spoken meetings. In *INTER-SPEECH*, pages 78–81.
- Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06*, pages 364–372, Stroudsburg, PA, USA. Association for Computational Linguistics.
- David Graff. 2003. English Gigaword.
- Michael Heilman and Noah A. Smith. 2010. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10*, pages 609–617, Stroudsburg, PA, USA. Association for Computational Linguistics.
- A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003. The icsi meeting corpus. volume 1, pages I–364–I–367 vol.1.
- Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning, ECML '98*, pages 137–142, London, UK, UK. Springer-Verlag.
- Thorsten Joachims. 1999. Advances in kernel methods. chapter Making large-scale support vector machine learning practical, pages 169–184. MIT Press, Cambridge, MA, USA.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03*, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ioannis Konstas and Mirella Lapata. 2012. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1, ACL '12*, pages 369–378, Stroudsburg, PA, USA. Association for Computational Linguistics.
- J R Landis and G G Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.
- Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10*, pages 912–920, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 71–78.
- Fei Liu and Yang Liu. 2009. From extractive to abstractive meeting summaries: can it be done by sentence compression? In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, ACLShort '09*, pages 261–264, Stroudsburg, PA, USA. Association for Computational Linguistics.

- I. Mccowan, G. Lathoud, M. Lincoln, A. Lisowska, W. Post, D. Reidsma, and P. Wellner. 2005. The ami meeting corpus. In *In: Proceedings Measuring Behavior 2005, 5th International Conference on Methods and Techniques in Behavioral Research*. L.P.J.J. Noldus, F. Grieco, L.W.S. Loijens and P.H. Zimmerman (Eds.), Wageningen: Noldus Information Technology.
- Gabriel Murray, Steve Renals, and Jean Carletta. 2005. Extractive summarization of meeting recordings. In *INTERSPEECH*, pages 593–596.
- Gabriel Murray, Giuseppe Carenini, and Raymond Ng. 2010a. Interpretation and transformation for abstracting conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 894–902, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Gabriel Murray, Giuseppe Carenini, and Raymond T. Ng. 2010b. Generating and validating abstracts of meeting conversations: a user study. In *INLG*.
- S. B. Needleman and C. D. Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, March.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge University Press, New York, NY, USA.
- Korbinian Riedhammer, Benoit Favre, and Dilek Hakkani-Tür. 2010. Long story short - global unsupervised models for keyphrase based meeting summarization. *Speech Commun.*, 52(10):801–815, October.
- Oana Sandu, Giuseppe Carenini, Gabriel Murray, and Raymond Ng. 2010. Domain adaptation to summarize human conversations. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, DANLP 2010, pages 16–22, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Alex J. Smola and Bernhard Schölkopf. 2004. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, August.
- Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP*, volume 2, pages 901–904, Denver, USA.
- Marilyn A. Walker, Owen Rambow, and Monica Rogati. 2001. Spot: a trainable sentence planner. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lu Wang and Claire Cardie. 2011. Summarizing decisions in spoken meetings. In *Proceedings of the Workshop on Automatic Summarization for Different Genres, Media, and Languages*, WASDGL '11, pages 16–24, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lu Wang and Claire Cardie. 2012. Focused meeting summarization via unsupervised relation extraction. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '12, pages 304–313, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lusheng Wang and Tao Jiang. 1994. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348.
- Shasha Xie, Yang Liu, and Hui Lin. 2008. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *in Proc. of IEEE Spoken Language Technology (SLT)*.

# A Statistical NLG Framework for Aggregated Planning and Realization

Ravi Kondadadi\*, Blake Howald and Frank Schilder

Thomson Reuters, Research & Development  
610 Opperman Drive, Eagan, MN 55123

firstname.lastname@thomsonreuters.com

## Abstract

We present a hybrid natural language generation (NLG) system that consolidates macro and micro planning and surface realization tasks into one statistical learning process. Our novel approach is based on deriving a template bank automatically from a corpus of texts from a target domain. First, we identify domain specific entity tags and Discourse Representation Structures on a per sentence basis. Each sentence is then organized into semantically similar groups (representing a domain specific concept) by  $k$ -means clustering. After this semi-automatic processing (human review of cluster assignments), a number of corpus-level statistics are compiled and used as features by a ranking SVM to develop model weights from a training corpus. At generation time, a set of input data, the collection of semantically organized templates, and the model weights are used to select optimal templates. Our system is evaluated with automatic, non-expert crowdsourced and expert evaluation metrics. We also introduce a novel automatic metric – *syntactic variability* – that represents linguistic variation as a measure of unique template sequences across a collection of automatically generated documents. The metrics for generated *weather* and *biography* texts fall within acceptable ranges. In sum, we argue that our statistical approach to NLG reduces the need for complicated knowledge-based architectures and readily adapts to different domains with reduced development time.

---

\*Ravi Kondadadi is now affiliated with Nuance Communications, Inc.

## 1 Introduction

NLG is the process of generating natural-sounding text from non-linguistic inputs. A typical NLG system contains three main components: (1) Document (Macro) Planning - deciding what content should be realized in the output and how it should be structured; (2) Sentence (Micro) planning - generating a detailed sentence specification and selecting appropriate referring expressions; and (3) Surface Realization - generating the final text after applying morphological modifications based on syntactic rules (*see e.g.*, Bateman and Zock (2003), Reiter and Dale (2000) and McKeown (1985)). However, document planning is arguably one of the most crucial components of an NLG system and is responsible for making the texts express the desired communicative goal in a coherent structure. If the document planning stage fails, the communicative goal of the generated text will not be met even if the other two stages are perfect. While most traditional systems simplify development by using a pipelined approach where (1-3) are executed in a sequence, this can result in errors at one stage propagating to successive stages (*see e.g.*, Robin and McKeown (1996)). We propose a hybrid framework that combines (1-3) by converting data to text in one single process.

Most NLG systems fall into two broad categories: knowledge-based and statistical. Knowledge-based systems heavily depend on having domain expertise to come up with hand-crafted rules at each stage of a pipeline. Although knowledge-based systems can produce high quality text, they are (1) very expensive to build, involving a lot of discussion with the end users of the system for the document planning stage alone; (2) have limited linguistic coverage, as it is time consuming to capture linguistic variation; and (3) one has to start from scratch for each new domain because the developed components cannot be reused.

Statistical systems, on the other hand, are fairly inexpensive, more adaptable and rely on having historical data for the given domain. Coverage is likely to be high if more historical data is available. The main disadvantage with statistical systems is that they are more prone to errors and the output text may not be coherent as there are less constraints on the generated text.

Our framework is a hybrid of statistical and template-based systems. Many knowledge-based systems use templates to generate text. A template structure contains “gaps” that are filled to generate the output. The idea is to create a lot of templates from the historical data and select the right template based on some constraints. To the best of our knowledge, this is the first hybrid statistical-template-based system that combines all three stages of NLG. Experiments with different variants of our system (for *biography* and *weather* subject matter domains) demonstrate that our system generates reasonable texts.

Also, in addition to the standard metrics used to evaluate NLG systems (e.g., BLEU, NIST, etc.), we present a unique text evaluation metric called *syntactic variability* to measure the linguistic variation of generated texts. This metric applies to the document collection level and is based on computing the number of unique template sequences among all the generated texts. A higher number indicates the texts are more variable and natural-sounding whereas a lower number shows they are more redundant. We argue that this metric is useful for evaluating template-based systems and for *any* type of text generation for domains where linguistic variability is favored (e.g., the user is expected to go through more than one document in the same session).

The main contributions of this paper are (1) A statistical NLG system that combines document and sentence planning and surface realization into one single process; and (2) A new metric – *syntactic variability* – is proposed to measure the syntactic and morphological variability of the generated texts. We believe this is the first work to propose an automatic metric to measure linguistic variability of generated texts in NLG.

Section 2 provides an overview of related work on NLG. We present our main system in Section 3. The system is evaluated and discussed in Section 4. Finally, we conclude in Section 5 and point out future directions of research.

## 2 Background

Typically, knowledge-based NLG systems are implemented by rules and, as mentioned above, have a pipelined architecture for the document and sentence planning stages and surface realization (Hovy, 1993; Moore and Paris, 1993). However, document planning is arguably the most important task (Sripada et al., 2001). It follows that approaches to document planning are rule-based as well and, concomitantly, are usually domain specific. For example, Bouayad-Agha, et al. (2011) proposed document planning based on an ontology knowledge base to generate football summaries. For rule-based systems, rules exist for selecting content to grammatical choices to post-processing (e.g., pronoun generation). These rules are often tailored to a given system, with input from multiple experts; consequently, there is a high associated development cost (e.g., 12 person months for the SUMTIME-METEO system (Belz, 2007)).

Statistical approaches can reduce extensive development time by relying on corpus data to “learn” rules for one or more components of an NLG system (Langkilde and Knight, 1998). For example, Duboue and McKeown (2003) proposed a statistical approach to extract content selection rules for biography descriptions. Further, statistical approaches should be more adaptable to different domains than their rule-based equivalents (Angeli et al., 2012). For example, Barzilay and Lapata (2005) formulated content selection as a classification task to produce football summaries and Kelly et al. (2009) extended Barzilay and Lapata’s approach for generating match reports for cricket.

The present work builds on Howald et al. (2013) where, in a given corpus, a combination of domain specific named entity tagging and clustering sentences (based on semantic predicates) were used to generate templates. However, while the system consolidated both sentence planning and surface realization with this approach (described in more detail in Section 3), the document plan was given via the input data and sequencing information was present in training documents. For the present research, we introduce a similar method that leverages the distributions of document-level features in the training corpus to incorporate a statistical document planning component. Consequently, we are able to create a streamlined statistical NLG architecture that balances natural

human-like variability with appropriate and accurate information.

### 3 Methodology

In order to generate text for a given domain our system runs input data through a statistical ranking model to select a sequence of templates that best fit the input data (E). In order to build the ranking model, our system takes historical data (corpus) for the domain through four components: (A) preprocessing; (B) “conceptual unit” creation; (C) collecting statistics; and (D) ranking model building (summarized in Figure 1). In this section, we describe each component in detail.

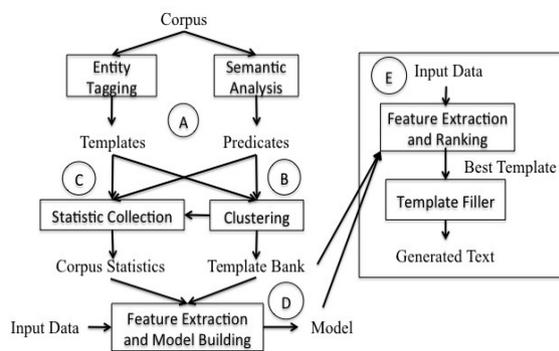


Figure 1: System Architecture.

#### 3.1 Preprocessing

The first component processes the given corpus to extract templates. We assume that each document in the corpus is classified to a specific domain. Preprocessing involves uncovering the underlying semantic structure of the corpus and using this as a foundation for template creation (Lu et al., 2009; Lu and Ng, 2011; Konstas and Lapata, 2012).

We first split each document in the corpus into sentences and create a shallow Discourse Representation Structure (following Discourse Representation Theory (Kamp and Reyle, 1993)) of each sentence. The DRS consists of semantic predicates and named entity tags. We use *Boxer* semantic analyzer (Bos, 2008) to extract semantic predicates such as EVENT or DATE. In parallel, domain specific named entity tags are identified and, in conjunction with the semantic predicates, are used to create templates. We developed the named-entity tagger for the *weather* domain ourselves. To tag entities in the *biography* domain, we used OpenCalais (www.opencalais.com). For example, in the biography in (1), the conceptual

meaning (semantic predicates and domain-specific entities) of sentences (a-b) are represented in (c-d). The corresponding templates are showing in (e-f).

##### (1) Sentence

- a. Mr. Mitsutaka Kambe has been serving as Managing Director of the 77 Bank, Ltd. since June 27, 2008.
- b. He holds a Bachelor’s in finance from USC and a MBA from UCLA.

##### Conceptual Meaning

- c. SERVING | TITLE | PERSON | COMPANY | DATE
- d. HOLDS | DEGREE | SUBJECT | INSTITUTION | EVENT

##### Templates

- e. [person] has been serving as [title] of the [company] since [date].
- f. [person] holds a [degree] in [subject] from [institution] and a [degree] from [institution].

The outputs of the preprocessing stage are the template bank and predicate information for each template in the corpus.<sup>1</sup>

#### 3.2 Creating Conceptual Units

The next step is to create conceptual units for the corpus by clustering templates. This is a semi-automatic process where we use the predicate information for each template to compute similarity between templates. We use *k*-means clustering with *k* (equivalent to the number of semantic concepts in the domain) set to an arbitrarily high value (100) to over-generate (using the WEKA toolkit (Witten and Frank, 2005)). This allows for easier manual verification of the generated clusters and we merge them if necessary. We assign a unique identifier called a CuId (Conceptual Unit Identifier) to each cluster, which represents a “conceptual unit”. We associate each template in the corpus to a corresponding CuId. For example, in (2), using the templates in (1e-f), the identified named entities are assigned to a clustered CuId (2a-b).

##### (2) Conceptual Units

- a. {CuId : 000} – [person] has been serving as [title] of the [company] since [date].
- b. {CuId : 001} – [person] holds a [degree] in [subject] from [institution] and a [degree] from [institution].

At this stage, we will have a set of conceptual units with corresponding template collections (see Howald et al. (2013) for a further explanation of Sections 3.1-3.2).

<sup>1</sup>A similar approach to the clustering of semantic content is found in Duboue and McKeown (2003), where text with stopwords removed were used as semantic input. Boxer provides a similar representation with the addition of domain general tags. However, to contrast our work from Duboue and McKeown, which focused on content selection, we are focused on learning templates from the semantic representations for the complete generation system (covering content selection, aggregation, sentence and document planning).

### 3.3 Collecting Corpus Statistics

After identifying the different conceptual units and the template bank, we collect a number of statistics from the corpus:

- Frequency distribution of templates overall and per position
- Frequency distribution of CuIds overall and per position
- Average number of entity tags by CuId as well as the entity distribution by CuId
- Average number of entity tags by position as well as the entity distribution by position
- Average number of words per CuId.
- Average number of words per CuId and position combination.
- Average number of words per position
- Frequency distribution of the main verbs by position
- Frequency distribution of CuId sequences (bigrams and trigrams only) overall and per position
- Frequency distribution of template sequences (bigrams and trigrams only) overall and per position
- Frequency distribution of entity tag sequences overall and per position
- The average, minimum, maximum number of CuIds across all documents

As discussed in the next section, these statistics are turned into features used for building a ranking model in the next component.

### 3.4 Building a ranking model

The core component of our system is a statistical model that ranks a set of templates for a given position (sentence 1, sentence 2, ..., sentence n) based on the input data. The input data in our tasks was extracted from a training document; this serves as a temporary surrogate to a database. The task is to learn the ranks of all the templates from all CuIds at each position.

To generate the training data, we first filter the templates that have named entity tags not specified in the input data. This will make sure the generated text does not have any unfilled entity tags. We then rank templates according to the Levenshtein edit distance (Levenshtein, 1966) from the template corresponding to the current sentence in the training document (using the top 10 ranked templates in training for ease of processing effort). We experimented with other ranking schemes such as entity-based similarity (similarity between entity sequences in the templates) and a combination of edit-distance based and entity-based similarities. We obtained better results with edit distance. For each template, we generate the following features to build the ranking model. Most of the features are based on the corpus statistics mentioned above.

- **CuId given position:** This is a binary feature where the current CuId is either the same as the most frequent CuId for the position (1) or not (0).
- **Overlap of named entities:** Number of common entities between current CuId and most likely CuId for the position
- **Prior template:** Probability of the sequence of templates selected at the previous position and the current template (iterated for the last three positions).
- **Prior CuId:** Probability of the sequence of the CuId selected at the previous position and the current CuId (iterated for the last three positions).
- **Difference in number of words:** Absolute difference between number of words for current template and average number of words for the CuId
- **Difference in number of words given position:** Absolute difference between number of words for current template and average number of words for CuId at given position
- **Percentage of unused data:** This feature represents the portion of the unused input so far.
- **Difference in number of named entities:** Absolute difference between the number of named entities in the current template and the average number of named entities for the current position
- **Most frequent verb for the position:** Binary valued feature where the main verb of the template belongs to the most frequent verb group given the position is either the same (1) or not (0).
- **Average number of words used:** Ratio of number of words in the generated text so far to the average number of words.
- **Average number of entities:** Ratio of number of named entities in the generated text so far to the average number of named entities.
- **Most likely CuId given position and previous CuId:** Binary feature indicating if the current CuId is most likely given the position and the previous CuId.
- **Similarity between the most likely template in CuId and current template:** Edit distance between the current template and the most likely template for the current CuId.
- **Similarity between the most likely template in CuId given position and current template:** Edit distance between the current template and the most likely template for the current CuId at the current position.

We used a linear kernel for a ranking SVM (Joachims, 2002) (*cost* set to total queries) to learn the weights associated with each feature for the different domains.

### 3.5 Generation

At generation time, our system has a set of input data, a semantically organized template bank (collection of templates organized by CuId) and a model from training on the documents for a given domain. We first filter out those templates that contain a named entity tag not present in the input data. Then, we compute a score for each of the remaining templates from the feature values and the feature weights from the model. The template with the highest overall score is selected and filled with matching entity tags from the input data and

appended to the generated text.

Before generating the next sentence, we track those entities used in the initial sentence generation and decide to either remove those entities from the input data or keep the entity for one or more additional sentence generations. For example, in the *biography* discourses, the name of the person may occur only once in the input data, but it may be useful for creating good texts to have that person's name available for subsequent generations. To illustrate in (3), if we remove *James Smithton* from the input data after the initial generation, an irrelevant sentence (d) is generated as the input data will only have one company after the removal of *James Smithton* and the model will only select a template with one company. If we keep *James Smithton*, then the generations in (a-b) are more cohesive.

(3) *Use more than once*

- a. Mr. James Smithton was appointed CFO at Fordway Internation in April.
- b. Previously, Mr. Smithton was CFO of the Keyes Development Group.

*Use once and remove*

- c. Mr. James Smithton was appointed CFO at Fordway Internation in April.
- d. Keyes Development Group is a venture capital firm.

Deciding on what type of entities and how to remove them is different for each domain. For example, some entities are very unique to a text and should not be made available for subsequent generations as doing so would lead to unwanted redundancies (*e.g.*, mentioning the name of current company in a biography discourse more than once as in (3)) and some entities are general and should be retained. Our system possesses the ability to monitor the data usage from historical data and we can set parameters (based on the distribution of entities) on the usage to ensure coherent generations for a given domain.

Once the input data has been modified (*i.e.*, an entity have been removed, replaced or retained), it serves as the new input data for the next sentence generation. This process repeats until reaching the minimum number of sentences for the domain (determined from the training corpus statistic) and then continues until all of the remaining input data is consumed (and not to exceed the pre-determined maximum number of sentences, also determined from the training corpus statistic).

## 4 Evaluation and Discussion

In this section, we first discuss the corpus data used to train and generate texts. Then, the results of both automatic and human evaluations of our system's generations against the original and baseline texts are considered as a means of determining performance. For all experiments reported in this section, the baseline system selects the most frequent conceptual unit at the given position, chooses the most likely template for the conceptual unit, and fills the template with input data. The above process is repeated until the number of sentences is less than or equal to the average number of sentences for the given domain.

### 4.1 Data

We ran our system on two different domains: corporate officer and director *biographies* and offshore oil rig *weather* reports from the SUMTIME-METEO corpus ((Reiter et al., 2005)). The *biography* domain includes 1150 texts ranging from 3-17 sentences and the *weather* domain includes 1045 weather reports ranging from 1-6 sentences.<sup>2</sup> We used a training-test(generation) split of 70/30.

(4) provides generation comparisons for the system (*DocSys*), baseline (*DocBase*) and original (*DocOrig*) randomly selected text snippets from each domain. The variability of the generated texts ranges from a close similarity to slightly shorter - not an uncommon (Belz and Reiter, 2006), but not necessarily detrimental, observation for NLG systems (van Deemter et al., 2005).

(4) *Weather.DocOrig*

- a. Another weak cold front will move ne to Cornwall by later Friday.  
*Weather.DocSys*
- b. Another weak cold front will move ne to Cornwall during Friday.  
*Weather.DocBase*
- c. Another weak cold front from ne through the Cornwall will remain slow moving.

*Bio.DocOrig*

- d. He previously served as Director of Sales Planning and Manager of Loan Center.  
*Bio.DocSys*
- e. He previously served as Director of Sales in Loan Center of the Company.  
*Bio.DocBase*

<sup>2</sup>The SUMTIME-METEO project is a common benchmark in NLG. However, we provide no comparison between our system and SUMTIME-METEO as our system utilized the generated forecasts from SUMTIME-METEO's system as the historical data. We cannot compare with other statistical generation systems like (Belz, 2007) as they only focussed on the part of the forecasts the predicts wind characteristics whereas our system generates the complete forecasts.

f. He previously served as Director of Sales of the Company.

The *DocSys* and *DocBase* generations are largely grammatical and coherent on the surface with some variance, but there are graded semantic variations (e.g., *Director of Sales Planning* vs. *Director of Sales* (4g-h) and *move ne to Cornwall* vs. *from ne through the Cornwall*). Both automatic and human evaluations are required in NLG to determine the impact of these variances on the understandability of the texts in general (non-experts) and as they are representative of particular subject matter domains (experts). The following sections discuss the evaluation results.

## 4.2 Automatic Metrics

We used BLEU-4 (Papineni et al., 2002), METEOR (v.1.3) (Denkowski and Lavie, 2011) to evaluate the texts at document level. Both BLEU-4 and METEOR originate from machine translation research. BLEU-4 measures the degree of 4-gram overlap between documents. METEOR uses a unigram weighted  $f$ -score less a penalty based on chunking dissimilarity. These metrics only evaluate the text on a document level but fail to identify “syntactic repetitiveness” across documents in a document collection. This is an important characteristic of a document collection to avoid banality. To address this issue, we propose a new automatic metric called *syntactic variability*. In order to compute this metric, each document should be represented as a sequence of templates by associating each sentence in the document with a template in the template bank. *Syntactic variability* is defined as the percentage of unique template sequences across all generated documents. It ranges between 0 and 1. A higher value indicates that more documents in the collection are linguistically different from each other and a value closer to zero shows that most of documents have the similar language despite different input data.<sup>3</sup>

As indicated in Figure 2, the BLEU-4 scores are low for all *DocSys* and *DocBase* generations (as compared to *DocOrig*) for each domain. However, the METEOR scores, while low overall (ranging from .201-.437) are noticeably increased over BLEU-4 (which ranges from .199-.320).

Given the nature of each metric, the results indicate that the generated and baseline texts have

<sup>3</sup>Of course, syntactic and semantic repetitiveness could be captured by *syntactic variability*, but only if this is the nature of the underlying historical data - *financial* texts tend to be fairly repetitive.

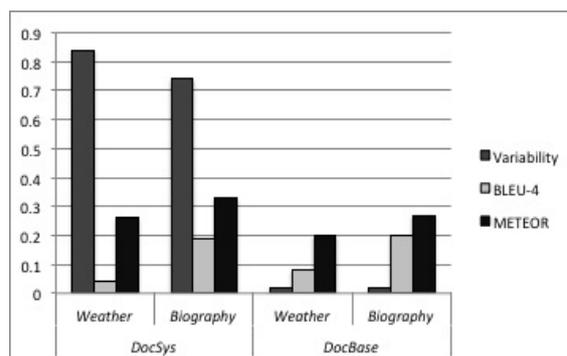


Figure 2: Automatic Evaluations.

very different surface realizations compared to the originals (low BLEU-4), but are still capturing the content of the originals (higher METEOR). Both BLEU-4 and METEOR measure the similarity of the generated text to the original text, but fail to penalize repetitiveness across texts, which is addressed by the *syntactic variability* metric. There is no statistically significant difference between *DocSys* and *DocBase* generations for METEOR and BLEU-4.<sup>4</sup> However, there is a statistically significant difference in the *syntactic variability* metric for both domains (*weather* -  $\chi^2=137.16$ , d.f.=1,  $p<.0001$ ; *biography* -  $\chi^2=96.641$ , d.f.=1,  $p<.0001$ ) - the variability of the *DocSys* generations is greater than the *DocBase* generations, which shows that texts generated by our system are more variable than the baseline texts.

The use of automatic metrics is a common evaluation method in NLG, but they must be reconciled against non-expert and expert level evaluations.

## 4.3 Non-Expert Human Evaluations

Two sets of crowdsourced human evaluation tasks (run on CrowdFlower) were constructed to compare against the automatic metrics: (1) an understandability evaluation of the entire text on a three-point scale: **Fluent** = no grammatical or informative barriers; **Understandable** = some grammatical or informative barriers; **Disfluent** = significant grammatical or informative barriers; and (2) a sentence-level preference between sentence pairs (e.g., “Do you prefer Sentence A (from *DocOrig*) or the corresponding Sentence B (from *DocBase/DocSys*)”).

<sup>4</sup>BLEU-4: *weather* -  $\chi^2=1.418$ , d.f.=1,  $p=.230$ ; *biography* -  $\chi^2=0.311$ , d.f.=1,  $p=.354$ . METEOR: *weather* -  $\chi^2=1.016$ , d.f.=1,  $p=.313$ ; *biography* -  $\chi^2=0.851$ , d.f.=1,  $p=.354$ .

Over 100 native English speakers contributed, each one restricted to providing no more than 50 responses and only after they successfully answered 4 “gold data” questions correctly. We also omitted those evaluators with a disproportionately high response rate. No other data was collected on the contributors (although geographic data (country, region, city) and IP addresses were available). For the sentence-level preference task, the pair orderings were randomized to prevent click bias.

For the text-understandability task, 40 documents were chosen at random from the *DocOrig* test set along with the corresponding 40 *DocSys* and *DocBase* generations (240 documents total/120 for each domain). 8 judgments per document were solicited from the crowd (1920 total judgments, 69.51 average agreement) and are summarized in Figures 3 and 4 (*biography* and *weather* respectively).

If the system is performing well and the ranking model is actually contributing to increased performance, the accepted trend should be that the *DocOrig* texts are more fluent and preferred compared to both the *DocSys* and *DocBase* systems. However, the differences between *DocOrig* and *DocSys* will not be significant, the differences between *DocOrig* and *DocBase* and *DocSys* and *DocBase* **will** be significantly different.

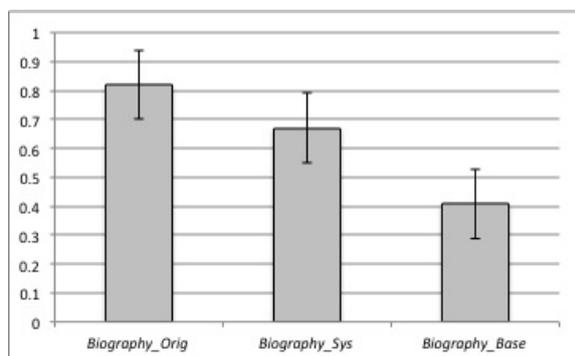


Figure 3: Biography Text Evaluations.

Focusing on fluency ratings, it is expected that the *DocOrig* generations will have the highest fluency (as they are human generated). Further, if the *DocSys* is performing well, it is expected that the fluency rating will be less than the *DocOrig* and higher than *DocBase*. Figure 3, which shows the *biography* text evaluations, demonstrates this acceptable distribution of performances.

For the *weather* discourses, as evident from Figure 4, the acceptable trend holds between the

*DocSys* and *DocBase* generations, and the *DocSys* generation fluency is actually slightly higher than *DocOrig*. This is possibly because the *DocOrig* texts are from a particular subject matter - weather forecasts for offshore oil rigs in the U.K. - which may be difficult for people in general to understand. Nonetheless, the demonstrated trend is favorable to our system.

In terms of significance, there are no statistically significant differences between the systems for *weather* (*DocOrig* vs. *DocSys* -  $\chi^2=0.347$ , d.f.=1,  $p=.555$ ; *DocOrig* vs. *DocBase* -  $\chi^2=0.090$ , d.f.=1,  $p=.764$ ; *DocSys* vs. *DocBase* -  $\chi^2=0.790$ , d.f.=1,  $p=.373$ ). While this is a good result for comparing *DocOrig* and *DocSys* generations, it is not for the other pairs. However, numerically, the trend is in the right direction despite not being able to demonstrate significance. For *biography*, the trend fits nicely both numerically and in terms of statistical significance (*DocOrig* vs. *DocSys* -  $\chi^2=5.094$ , d.f.=1,  $p=.024$ ; *DocOrig* vs. *DocBase* -  $\chi^2=35.171$ , d.f.=1,  $p<.0001$ ; *DocSys* vs. *DocBase* -  $\chi^2=14.000$ , d.f.=1,  $p<.0001$ ).

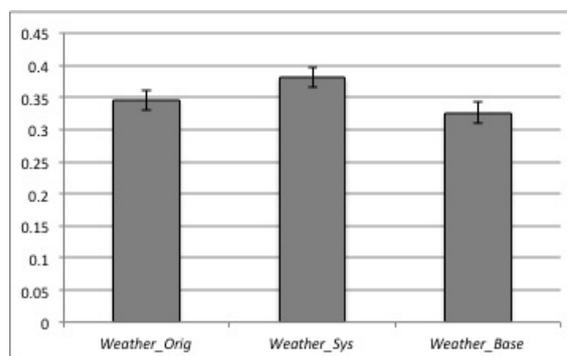


Figure 4: Weather Text Evaluations.

For the sentence preference task, equivalent sentences across the 120 documents were chosen at random (80 sentences from *biography* and 74 sentences from *weather*). 8 judgments per comparison were solicited from the crowd (3758 total judgments, 75.87 average agreement) and are summarized in Figures 5 and 6 (*biography* and *weather*, respectively).

Similar to the text-understandability task, an acceptable performance pattern should include the *DocOrig* texts being preferred to both *DocSys* and *DocBase* generations and the *DocSys* generation preferred to the *DocBase*. The closer the *DocSys* generation is to the *DocOrig*, the better *DocSys* is performing. The *biography* domain illus-

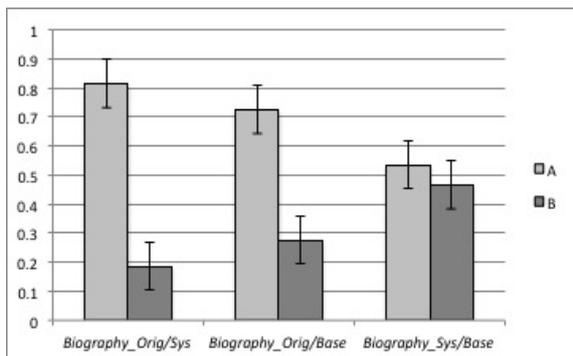


Figure 5: Biography Sentence Evaluations.

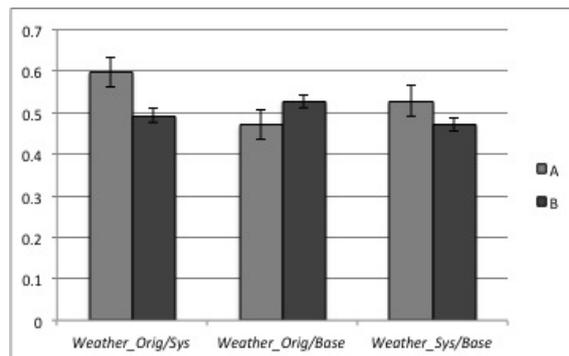


Figure 6: Weather Sentence Evaluations.

trates this scenario (Figure 5) where the results are similar to the text-understandability experiments. In contrast, for *weather* domain, sentences from *DocBase* system were preferred to our system's (Figure 6). We looked at the cases where the preferences were in favor of *DocBase*. It appears that because of high *syntactic variability*, our system can produce quite complex sentences where as the non-experts seem to prefer shorter and simpler sentences because of the complexity of the text.

In terms of significance, there are no statistically significant differences between the systems for *weather* (*DocOrig* vs. *DocSys* -  $\chi^2=6.48$ , d.f.=1,  $p=.011$ ; *DocOrig* vs. *DocBase* -  $\chi^2=.720$ , d.f.=1,  $p=.396$ ; *DocSys* vs. *DocBase* -  $\chi^2=.720$ , d.f.=1,  $p=.396$ ). The trend is different compared to the fluency metric above in that the *DocBase* system is outperforming the *DocOrig* generations to an almost statistically significant difference - the remaining comparisons follow the trend. We believe that this is for similar reasons stated above - i.e., the generation may be a more digestible version of a technical document. More problematic is the results of the *biography* evaluations. Here there is a statistically significant difference between the *DocSys* and *DocOrig* and no statistically significant difference between the *DocSys* and *DocBase* generations (*DocOrig* vs. *DocSys* -  $\chi^2=76.880$ , d.f.=1,  $p<.0001$ ; *DocOrig* vs. *DocBase* -  $\chi^2=38.720$ , d.f.=1,  $p<.0001$ ; *DocSys* vs. *DocBase* -  $\chi^2=.720$ , d.f.=1,  $p=.396$ ). Again, this distribution of preferences is numerically similar to the trend we would like to see, but the statistical significance indicates that there is some ground to make up. Expert evaluations are potentially informative for identifying specific shortcomings and how best to address them.

#### 4.4 Expert Human Evaluations

We performed expert evaluations for the *biography* domain only as we do not have access to weather experts. The four *biography* reviewers are journalists who write short biographies for news archives.

For the *biography* domain, evaluations of the texts were largely similar to the evaluations of the non-expert crowd (76.22 average agreement for the sentence-preference task and 72.95 for the text-understandability task). For example, the **disfluent** ratings were highest for the *DocBase* generations and lowest for the *DocOrig* generations. Also, the **fluent** ratings were highest for the *DocOrig* generations, and while the combined **fluent** and **understandable** are higher for *DocSys* as compared to *DocBase*, the *DocBase* generations had a 10% higher **fluent** score (58.22%) as compared to the *DocSys* **fluent** score (47.97%). Based on notes from the reviewers, the succinctness of the the *DocBase* generations are preferred in some ways as they are in keeping with certain editorial standards. This is further reflected in the sentence preferences being 70% in favor of the *DocBase* generations as compared to the *DocSys* generations (all other sentence comparisons were consistent with the non-expert crowd).

These expert evaluations provide much needed clarity to the NLG process. Overall, our system is generating clearly acceptable texts. Further, there are enough parameters inherent in the system to tune to different domain expectations. This is an encouraging result considering that no experts were involved in the development of the system - a key contrast to many other existing (especially rule-based) NLG systems.

## 5 Conclusions and Future Work

We have presented a hybrid (template-based and statistical), single-staged NLG system that generates natural sounding texts and is domain-adaptable. Our experiments with both experts and non-experts demonstrate that the system-generated texts are comparable to human-authored texts. The development time to adapt our system to new domains is small compared to other NLG systems; around a week to adapt the system to *weather* and *biography* domains. Most of the development time was spent on creating the domain-specific entity taggers for the *weather* domain. The development time would be reduced to hours if the historical data for a domain is readily available with the corresponding input data.

The main limitation of our system is that it requires significant historical data. Our system does consolidate many traditional components (macro and micro-planning, lexical choice and aggregation),<sup>5</sup> but the system cannot be applied to the domains with no historical data. The quality and the linguistic variability of the generated text is directly proportional to the amount of historical data available.

We also presented a new automatic metric to evaluate generated texts at document collection level to identify boilerplate texts. This metric computes “syntactic repetitiveness” by counting the number of unique template sequences across the given document collection.

Future work will focus on extending our framework by adding additional features to the model that could improve the quality of the generated text. For example, most NLG pipelines have a separate component responsible for referring expression generation (Krahmer and van Deemter, 2012). While we address the associated concern of data consumption in Section 3.5, we currently do not have any features that would handle referring expression generation. We believe that this is possible by identifying referring expressions in templates and adding features to the model to give higher scores to the templates having relevant referring expressions. We also would like to investigate using all the top-scored templates instead of the highest-scoring template. This would help achieve better syntactic-variability scores by producing more natural-sounding texts.

<sup>5</sup>Lexical choice and aggregation are “handled” insofar as their existence in the historical data.

## Acknowledgments

This research is made possible by Thomson Reuters Global Resources (TRGR) with particular thanks to Peter Pircher, Jaclyn Sprtel and Ben Hachey for significant support. Thank you also to Khalid Al-Kofahi for encouragement, Leszek Michalak and Andrew Lipstein for expert evaluations and three anonymous reviewers for constructive feedback.

## References

- Gabor Angeli, Percy Liang, and Dan Klein. 2012. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods for Natural Language Processing (EMNLP 2010)*, pages 502–512.
- Regina Barzilay and Mirella Lapata. 2005. Collective content selection for concept-to-text generation. In *Proceedings of the 2005 Conference on Empirical Methods for Natural Language Processing (EMNLP 2005)*, pages 331–338.
- John Bateman and Michael Zock. 2003. Natural language generation. In R. Mitkov, editor, *Oxford Handbook of Computational Linguistics*, Research in Computational Semantics, pages 284–304. Oxford University Press, Oxford.
- Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *Proceedings of the European Association for Computational Linguistics (EACL’06)*, pages 313–320.
- Anja Belz. 2007. Probabilistic generation of weather forecast texts. In *Proceedings of Human Language Technologies 2007: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT’07)*, pages 164–171.
- Johan Bos. 2008. Wide-coverage semantic analysis with *Boxer*. In J. Bos and R. Delmonte, editors, *Semantics in Text Processing. STEP 2008 Conference Proceedings*, volume 1 of *Research in Computational Semantics*, pages 277–286. College Publications.
- Nadjet Bouayad-Agha, Gerard Casamayor, and Leo Wanner. 2011. Content selection from an ontology-based knowledge base for the generation of football summaries. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, pages 72–81.
- Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*, pages 85–91.

- Pablo A. Duboue and Kathleen R. McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 Conference on Empirical Methods for Natural Language Processing (EMNLP 2003)*, pages 2003–2007.
- Eduard H. Hovy. 1993. Automated discourse generation using discourse structure relations. *Artificial Intelligence*, 63:341–385.
- Blake Howald, Ravi Kondadadi, and Frank Schilder. 2013. Domain adaptable semantic clustering in statistical nlg. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, pages 143–154. Association for Computational Linguistics, March.
- Thorsten Joachims. 2002. *Learning to Classify Text Using Support Vector Machines*. Kluwer.
- Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Kluwer, Dordrecht.
- Colin Kelly, Ann Copestake, and Nikiforos Karamanis. 2009. Investigating content selection for language generation using machine learning. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG)*, pages 130–137.
- Ioannis Konstas and Mirella Lapata. 2012. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 369–378.
- Emiel Krahmer and Kees van Deemter. 2012. Computational generation of referring expression: A survey. *Computational Linguistics*, 38(1):173–218.
- Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics (ACL'98)*, pages 704–710.
- Vladimir Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710.
- Wei Lu and Hwee Tou Ng. 2011. A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. In *Proceedings of the 2011 Conference on Empirical Methods for Natural Language Processing (EMNLP 2011)*, pages 1611–1622.
- Wei Lu, Hwee Tou Ng, and Wee Sun Lee. 2009. Natural language generation with tree conditional random fields. In *Proceedings of the 2009 Conference on Empirical Methods for Natural Language Processing (EMNLP 2009)*, pages 400–409.
- Kathleen R. McKeown. 1985. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press.
- Johanna D. Moore and Cecile L. Paris. 1993. Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational Linguistics*, 19(4):651–694.
- Kishore Papineni, Slim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 311–318.
- Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press.
- Ehud Reiter, Somayajulu Sripada, Jim Hunter, and Jin Yu. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167:137–169.
- Jacques Robin and Kathy McKeown. 1996. Empirically designing and evaluating a new revision-based model for summary generation. *Artificial Intelligence*, 85(1-2).
- Somayajulu Sripada, Ehud Reiter, Jim Hunter, and Jin Yu. 2001. A two-stage model for content determination. In *Proceedings of the 8th European Workshop on Natural Language Generation (ENLG)*, pages 1–8.
- Kees van Deemter, Mariët Theune, and Emiel Krahmer. 2005. Real vs. template-based natural language generation: a false opposition? *Computational Linguistics*, 31(1):15–24.
- Ian Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Techniques with Java Implementation (2nd Ed.)*. Morgan Kaufmann, San Francisco, CA.

# Models of Translation Competitions

Mark Hopkins and Jonathan May

SDL Research

6060 Center Drive, Suite 150

Los Angeles, CA 90045

{mhopkins, jmay}@sdl.com

## Abstract

What do we want to learn from a translation competition and how do we learn it with confidence? We argue that a disproportionate focus on ranking competition participants has led to lots of different rankings, but little insight about which rankings we should trust. In response, we provide the first framework that allows an *empirical* comparison of different analyses of competition results. We then use this framework to compare several analytical models on data from the Workshop on Machine Translation (WMT).

## 1 The WMT Translation Competition

Every year, the Workshop on Machine Translation (WMT) conducts a competition between machine translation systems. The WMT organizers invite research groups to submit translation systems in eight different tracks: Czech to/from English, French to/from English, German to/from English, and Spanish to/from English.

For each track, the organizers also assemble a panel of judges, typically machine translation specialists.<sup>1</sup> The role of a judge is to repeatedly rank five different translations of the same source text. Ties are permitted. In Table 1, we show an example<sup>2</sup> where a judge (we’ll call him “jdoe”) has ranked five translations of the French sentence “Il ne va pas.”

Each such elicitation encodes ten pairwise comparisons, as shown in Table 2. For each competition track, WMT typically elicits between 5000 and 20000 comparisons. Once the elicitation process is complete, WMT faces a large database of comparisons and a question that must be answered: whose system is the best?

<sup>1</sup>Although in recent competitions, some of the judging has also been crowdsourced (Callison-Burch et al., 2010).

<sup>2</sup>The example does not use actual system output.

rank	system	translation
1	bbn	“He does not go.”
2 (tie)	uedin	“He goes not.”
2 (tie)	jhu	“He did not go.”
4	cmu	“He go not.”
5	kit	“He not go.”

Table 1: WMT elicits preferences by asking judges to simultaneously rank five translations, with ties permitted. In this (fictional) example, the source sentence is the French “Il ne va pas.”

source text	sys1	sys2	judge	preference
“Il ne va pas.”	bbn	cmu	jdoe	1
“Il ne va pas.”	bbn	jhu	jdoe	1
“Il ne va pas.”	bbn	kit	jdoe	1
“Il ne va pas.”	bbn	uedin	jdoe	1
“Il ne va pas.”	cmu	jhu	jdoe	2
“Il ne va pas.”	cmu	kit	jdoe	1
“Il ne va pas.”	cmu	uedin	jdoe	2
“Il ne va pas.”	jhu	kit	jdoe	1
“Il ne va pas.”	jhu	uedin	jdoe	0
“Il ne va pas.”	kit	uedin	jdoe	2

Table 2: Pairwise comparisons encoded by Table 1. A preference of 0 means neither translation was preferred. Otherwise the preference specifies the preferred system.

## 2 A Ranking Problem

For several years, WMT used the following heuristic for ranking the translation systems:

$$\text{ORIGWMT}(s) = \frac{\text{win}(s) + \text{tie}(s)}{\text{win}(s) + \text{tie}(s) + \text{loss}(s)}$$

For system  $s$ ,  $\text{win}(s)$  is the number of pairwise comparisons in which  $s$  was preferred,  $\text{loss}(s)$  is the number of comparisons in which  $s$  was dispreferred, and  $\text{tie}(s)$  is the number of comparisons in which  $s$  participated but neither system was preferred.

Recently, (Bojar et al., 2011) questioned the adequacy of this heuristic through the following ar-

gument. Consider a competition with systems  $A$  and  $B$ . Suppose that the systems are different but equally good, such that one third of the time  $A$  is judged better than  $B$ , one third of the time  $B$  is judged better than  $A$ , and one third of the time they are judged to be equal. The expected values of  $\text{ORIGWMT}(A)$  and  $\text{ORIGWMT}(B)$  are both  $2/3$ , so the heuristic accurately judges the systems to be equivalently good. Suppose however that we had duplicated  $B$  and had submitted it to the competition a second time as system  $C$ . Since  $B$  and  $C$  produce identical translations, they should always tie with one another. The expected value of  $\text{ORIGWMT}(A)$  would not change, but the expected value of  $\text{ORIGWMT}(B)$  would increase to  $5/6$ , buoyed by its ties with system  $C$ .

This vulnerability prompted (Bojar et al., 2011) to offer the following revision:

$$\text{BOJAR}(s) = \frac{\text{win}(s)}{\text{win}(s) + \text{loss}(s)}$$

The following year, it was BOJAR’s turn to be criticized, this time by (Lopez, 2012):

Superficially, this appears to be an improvement...couldn’t a system still be penalized simply by being compared to [good systems] more frequently than its competitors? On the other hand, couldn’t a system be rewarded simply by being compared against a bad system more frequently than its competitors?

Lopez’s concern, while reasonable, is less obviously damning than (Bojar et al., 2011)’s criticism of ORIGWMT. It depends on whether the collected set of comparisons is small enough or biased enough to make the variance in competition significant. While this hypothesis is plausible, Lopez makes no attempt to verify it. Instead, he offers a ranking heuristic of his own, based on a Minimum Feedback Arc solver.

The proliferation of ranking heuristics continued from there. The WMT 2012 organizers (Callison-Burch et al., 2012) took Lopez’s ranking scheme and provided a variant called Most Probable Ranking. Then, noting some potential pitfalls with that, they created two more, called Monte Carlo Playoffs and Expected Wins. While one could raise philosophical objections about each of these, where would it end? Ultimately, the WMT 2012 findings presented five different rankings for

the English-German competition track, with no guidance about which ranking we should pay attention to. How can we know whether one ranking is better than other? Or is this even the right question to ask?

### 3 A Problem with Rankings

Suppose four systems participate in a translation competition. Three of these systems are extremely close in quality. We’ll call these close1, close2, and close3. Nevertheless, close1 is very slightly better<sup>3</sup> than close2, and close2 is very slightly better than close3. The fourth system, called terrific, is a really terrific system that far exceeds the other three.

Now which is the better ranking?

terrific, close3, close1, close2 (1)

close1, terrific, close2, close3 (2)

Spearman’s rho<sup>4</sup> would favor the second ranking, since it is a less disruptive permutation of the gold ranking. But intuition favors the first. While its mistakes are minor, the second ranking makes the hard-to-forgive mistake of placing close1 ahead of the terrific system.

The problem is not with Spearman’s rho. The problem is the disconnect between the knowledge that we want a ranking to reflect and the knowledge that a ranking actually contains. Without this additional knowledge, we cannot determine whether one ranking is better than another, even if we know the gold ranking. We need to determine what information they lack, and define more rigorously what we hope to learn from a translation competition.

### 4 From Rankings to Relative Ability

Ostensibly the purpose of a translation competition is to determine the relative ability of a set of translation systems. Let  $\mathcal{S}$  be the space of all translation systems. Hereafter, we will refer to  $\mathcal{S}$  as the space of *students*. We choose this term to evoke the metaphor of a translation competition as a standardized test, which shares the same goal: to assess the relative abilities of a set of participants.

But what exactly do we mean by “ability”? Before formally defining this term, first recognize that it means little without context, namely:

<sup>3</sup>What does “better” mean? We’ll return to this question.

<sup>4</sup>Or Pearson’s correlation coefficient.

1. **What kind of source text do we want the systems to translate well?** Say system A is great at translating travel-related documents, but terrible at translating newswire. Meanwhile, system B is pretty good at both. The question “which system is better?” requires us to state how much we care about travel versus newswire documents – otherwise the question is underspecified.
2. **Who are we trying to impress?** While it’s tempting to think that translation quality is a universal notion, the 50-60% interannotator agreement in WMT evaluations (Callison-Burch et al., 2012) suggests otherwise. It’s also easy to imagine reasons why one group of judges might have different priorities than another. Think a Fortune 500 company versus web forum users. Lawyers versus laymen. Non-native versus native speakers. Posteditors versus Google Translate users. Different groups have different uses for translation, and therefore different definitions of what “better” means.

With this in mind, let’s define some additional elements of a translation competition. Let  $\mathcal{X}$  be the space of all possible segments of source text,  $\mathcal{J}$  be the space of all possible judges, and  $\Pi = \{0, 1, 2\}$  be the space of pairwise preferences.<sup>5</sup> We assume all spaces are countable. Unless stated otherwise, variables  $s_1$  and  $s_2$  represent students from  $\mathcal{S}$ , variable  $x$  represents a segment from  $\mathcal{X}$ , variable  $j$  represents a judge from  $\mathcal{J}$ , and variable  $\pi$  represents a preference from  $\Pi$ . Moreover, define the *negation*  $\hat{\pi}$  of preference  $\pi$  such that  $\hat{\pi} = 2$  (if  $\pi = 1$ ),  $\hat{\pi} = 1$  (if  $\pi = 2$ ), and  $\hat{\pi} = 0$  (if  $\pi = 0$ ).

Now assume a joint distribution  $P(s_1, s_2, x, j, \pi)$  specifying the probability that we ask judge  $j$  to evaluate students  $s_1$  and  $s_2$ ’s respective translations of source text  $x$ , and that judge  $j$ ’s preference is  $\pi$ . We will further assume that the choice of student pair, source text, and judge are marginally independent of one another. In other words:

$$\begin{aligned}
& P(s_1, s_2, x, j, \pi) \\
&= P(\pi|s_1, s_2, x, j) \cdot P(x|s_1, s_2, j) \\
&\quad \cdot P(j|s_1, s_2) \cdot P(s_1, s_2) \\
&= P(\pi|s_1, s_2, x, j) \cdot P(x) \cdot P(j) \cdot P(s_1, s_2) \\
&= P_{\mathcal{X}}(x) \cdot P_{\mathcal{J}}(j) \cdot P(s_1, s_2) \cdot P(\pi|s_1, s_2, x, j)
\end{aligned}$$

<sup>5</sup>As a reminder, 0 indicates no preference.

It will be useful to reserve notation  $P_{\mathcal{X}}$  and  $P_{\mathcal{J}}$  for the marginal distributions over source text and judges. We can marginalize over the source segments and judges to obtain a useful quantity:

$$\begin{aligned}
& P(\pi|s_1, s_2) \\
&= \sum_{x \in \mathcal{X}} \sum_{j \in \mathcal{J}} P_{\mathcal{X}}(x) \cdot P_{\mathcal{J}}(j) \cdot P(\pi|s_1, s_2, x, j)
\end{aligned}$$

We refer to this as the  $\langle P_{\mathcal{X}}, P_{\mathcal{J}} \rangle$ -*relative ability* of students  $s_1$  and  $s_2$ . By using different marginal distributions  $P_{\mathcal{X}}$ , we can specify what kinds of source text interest us (for instance,  $P_{\mathcal{X}}$  could focus most of its probability mass on German tweets). Similarly, by using different marginal distributions  $P_{\mathcal{J}}$ , we can specify what judges we want to impress (for instance,  $P_{\mathcal{J}}$  could focus all of its mass on one important corporate customer or evenly among all fluent bilingual speakers of a language pair).

With this machinery, we can express the purpose of a translation competition more clearly: to estimate the  $\langle P_{\mathcal{X}}, P_{\mathcal{J}} \rangle$ -*relative ability* of a set of students. In the case of WMT,  $P_{\mathcal{J}}$  presumably<sup>6</sup> defines a space of competent source-to-target bilingual speakers, while  $P_{\mathcal{X}}$  defines a space of newswire documents.

We’ll refer to an estimate of  $P(\pi|s_1, s_2)$  as a *preference model*. In other words, a preference model is a distribution  $Q(\pi|s_1, s_2)$ . Given a set of pairwise comparisons (e.g., Table 2), the challenge is to estimate a preference model  $Q(\pi|s_1, s_2)$  such that  $Q$  is “close” to  $P$ . For measuring distributional proximity, a natural choice is KL-divergence (Kullback and Leibler, 1951), but we cannot use it here because  $P$  is unknown.

Fortunately, if we have i.i.d. data drawn from  $P$ , then we can do the next best thing and compute the perplexity of preference model  $Q$  on this heldout test data. Let  $\mathcal{D}$  be a sequence of triples  $\langle s_1, s_2, \pi \rangle$  where the preferences  $\pi$  are i.i.d. samples from  $P(\pi|s_1, s_2)$ . The perplexity of preference model  $Q$  on test data  $\mathcal{D}$  is:

$$\text{perplexity}(Q|\mathcal{D}) = 2^{-\sum_{\langle s_1, s_2, \pi \rangle \in \mathcal{D}} \frac{1}{|\mathcal{D}|} \log_2 Q(\pi|s_1, s_2)}$$

How do we obtain such a test set from competition data? Recall that a WMT competition produces pairwise comparisons like those in Table 2.

<sup>6</sup>One could argue that it specifies a space of machine translation specialists, but likely these individuals are thought to be a representative sample of a broader community.

Let  $\mathcal{C}$  be the set of *comparisons*  $\langle s_1, s_2, x, j, \pi \rangle$  obtained from a translation competition. Competition data  $\mathcal{C}$  is not necessarily<sup>7</sup> sampled i.i.d. from  $P(s_1, s_2, x, j, \pi)$  because we may intentionally<sup>8</sup> bias data collection towards certain students, judges or source text. Also, because WMT elicits its data in batches (see Table 1), every segment  $x$  of source text appears in at least ten comparisons.

To create an appropriately-sized test set that closely resembles i.i.d. data, we isolate the subset  $\mathcal{C}'$  of comparisons whose source text appears in at most  $k$  comparisons, where  $k$  is the smallest positive integer such that  $|\mathcal{C}'| \geq 2000$ . We then create the test set  $\mathcal{D}$  from  $\mathcal{C}'$ :

$$\mathcal{D} = \{\langle s_1, s_2, \pi \rangle \mid \langle s_1, s_2, x, j, \pi \rangle \in \mathcal{C}'\}$$

We reserve the remaining comparisons for training preference models. Table 3 shows the resulting dataset sizes for each competition track.

Unlike with raw rankings, the claim that one preference model is better than another has testable implications. Given two competing models, we can train them on the same comparisons, and compare their perplexities on the test set. This gives us a quantitative<sup>9</sup> answer to the question of which is the better model. We can then publish a system ranking based on the most trustworthy preference model.

## 5 Baselines

Let's begin then, and create some simple preference models to serve as baselines.

### 5.1 Uniform

The simplest preference model is a uniform distribution over preferences, for any choice of students  $s_1, s_2$ :

$$Q(\pi|s_1, s_2) = \frac{1}{3} \quad \forall \pi \in \Pi$$

This will be our only model that does not require training data, and its perplexity on any test set will be 3 (i.e. equal to number of possible preferences).

### 5.2 Adjusted Uniform

Now suppose we have a set  $\mathcal{C}$  of comparisons available for training. Let  $\mathcal{C}_\pi \subseteq \mathcal{C}$  denote the subset of comparisons with preference  $\pi$ , and let

<sup>7</sup>In WMT, it certainly is not.

<sup>8</sup>To collect judge agreement statistics, for instance.

<sup>9</sup>As opposed to philosophical.

$\mathcal{C}(s_1, s_2)$  denote the subset comparing students  $s_1$  and  $s_2$ .

Perhaps the simplest thing we can do with the training data is to estimate the probability of ties (i.e. preference 0). We can then distribute the remaining probability mass uniformly among the other two preferences:

$$Q(\pi|s_1, s_2) = \begin{cases} \frac{|\mathcal{C}_0|}{|\mathcal{C}|} & \text{if } \pi = 0 \\ \frac{1 - \frac{|\mathcal{C}_0|}{|\mathcal{C}|}}{2} & \text{otherwise} \end{cases}$$

## 6 Simple Bayesian Models

### 6.1 Independent Pairs

Another simple model is the direct estimation of each relative ability  $P(\pi|s_1, s_2)$  independently. In other words, for each pair of students  $s_1$  and  $s_2$ , we estimate a separate preference distribution. The maximum likelihood estimate of each distribution would be:

$$Q(\pi|s_1, s_2) = \frac{|\mathcal{C}_\pi(s_1, s_2)| + |\mathcal{C}_{\hat{\pi}}(s_2, s_1)|}{|\mathcal{C}(s_1, s_2)| + |\mathcal{C}(s_2, s_1)|}$$

However the maximum likelihood estimate would test poorly, since any zero probability estimates for test set preferences would result in infinite perplexity. To make this model practical, we assume a symmetric Dirichlet prior with strength  $\alpha$  for each preference distribution. This gives us the following Bayesian estimate:

$$Q(\pi|s_1, s_2) = \frac{\alpha + |\mathcal{C}_\pi(s_1, s_2)| + |\mathcal{C}_{\hat{\pi}}(s_2, s_1)|}{3\alpha + |\mathcal{C}(s_1, s_2)| + |\mathcal{C}(s_2, s_1)|}$$

We call this the Independent Pairs preference model.

### 6.2 Independent Students

The Independent Pairs model makes a strong independence assumption. It assumes that even if we know that student A is much better than student B, and that student B is much better than student C, we can infer nothing about how student A will fare versus student C. Instead of directly estimating the relative ability  $P(\pi|s_1, s_2)$  of students  $s_1$  and  $s_2$ , we could instead try to estimate the *universal ability*  $P(\pi|s_1) = \sum_{s_2 \in \mathcal{S}} P(\pi|s_1, s_2) \cdot P(s_2|s_1)$  of each individual student  $s_1$  and then try to reconstruct the relative abilities from these estimates.

For the same reasons as before, we assume a symmetric Dirichlet prior with strength  $\alpha$  for each

preference distribution, which gives us the following Bayesian estimate:

$$Q(\pi|s_1) = \frac{\alpha + \sum_{s_2 \in \mathcal{S}} |\mathcal{C}_\pi(s_1, s_2)| + |\mathcal{C}_{\hat{\pi}}(s_2, s_1)|}{3\alpha + \sum_{s_2 \in \mathcal{S}} |\mathcal{C}(s_1, s_2)| + |\mathcal{C}(s_2, s_1)|}$$

The estimates  $Q(\pi|s_1)$  do not yet constitute a preference model. A downside of this approach is that there is no principled way to reconstruct a preference model from the universal ability estimates. We experiment with three ad-hoc reconstructions. The *asymmetric* reconstruction simply ignores any information we have about student  $s_2$ :

$$Q(\pi|s_1, s_2) = Q(\pi|s_1)$$

The *arithmetic* and *geometric* reconstructions compute an arithmetic/geometric average of the two universal abilities:

$$Q(\pi|s_1, s_2) = \frac{Q(\pi|s_1) + Q(\hat{\pi}|s_2)}{2}$$

$$Q(\pi|s_1, s_2) = [Q(\pi|s_1) * Q(\hat{\pi}|s_2)]^{\frac{1}{2}}$$

We respectively call these the (Asymmetric/Arithmetic/Geometric) Independent Students preference models. Notice the similarities between the universal ability estimates  $Q(\pi|s_1)$  and the BOJAR ranking heuristic. These three models are our attempt to render the BOJAR heuristic as preference models.

## 7 Item-Response Theoretic (IRT) Models

Let's revisit (Lopez, 2012)'s objection to the BOJAR ranking heuristic: "...couldn't a system still be penalized simply by being compared to [good systems] more frequently than its competitors?" The official WMT 2012 findings (Callison-Burch et al., 2012) echoes this concern in justifying the exclusion of reference translations from the 2012 competition:

[W]orkers have a very clear preference for reference translations, so including them unduly penalized systems that, through (un)luck of the draw, were pitted against the references more often.

Presuming the students are paired uniformly at random, this issue diminishes as more comparisons are elicited. But preference elicitation is expensive, so it makes sense to assess the relative ability of the students with as few elicitations as possible. Still, WMT 2012's decision to eliminate

references entirely is a bit of a draconian measure, a treatment of the symptom rather than the (perceived) disease. If our models cannot function in the presence of training data variation, then we should change the models, not the data. A model that only works when the students are all about the same level is not one we should rely on.

We experiment with a simple model that relaxes some independence assumptions made by previous models, in order to allow training data variation (e.g. who a student has been paired with) to influence the estimation of the student abilities. Figure 1(left) shows plate notation (Koller and Friedman, 2009) for the model's independence structure. First, each student's *ability* distribution is drawn from a common prior distribution. Then a number of translation *items* are generated. Each *item* is *authored* by a student and has a *quality* drawn from the student's *ability* distribution. Then a number of pairwise *comparisons* are generated. Each *comparison* has two *options*, each a translation *item*. The *quality* of each item is *observed* by a judge (possibly noisily) and then the judge states a *preference* by comparing the two *observations*.

We investigate two parameterizations of this model: Gaussian and categorical. Figure 1(right) shows an example of the Gaussian parameterization. The student ability distributions are Gaussians with a known standard deviation  $\sigma_a$ , drawn from a zero-mean Gaussian prior with known standard deviation  $\sigma_0$ . In the example, we show the ability distributions for students 6 (an above-average student, whose mean is 0.4) and 14 (a poor student, whose mean is -0.6). We also show an item authored by each student. Item 43 has a somewhat low quality of -0.3 (drawn from student 14's ability distribution), while item 205 is not student 6's best work (he produces a mean quality of 0.4), but still has a decent quality at 0.2. Comparison 1 pits these items against one another. A judge draws noise from a zero-mean Gaussian with known standard deviation  $\sigma_{obs}$ , then adds this to the item's actual quality to get an observed quality. For the first option (item 43), the judge draws a noise of -0.12 to observe a quality of -0.42 (worse than it actually is). For the second option (item 205), the judge draws a noise of 0.15 to observe a quality of 0.35 (better than it actually is). Finally, the judge compares the two observed qualities. If the absolute difference is lower than his decision

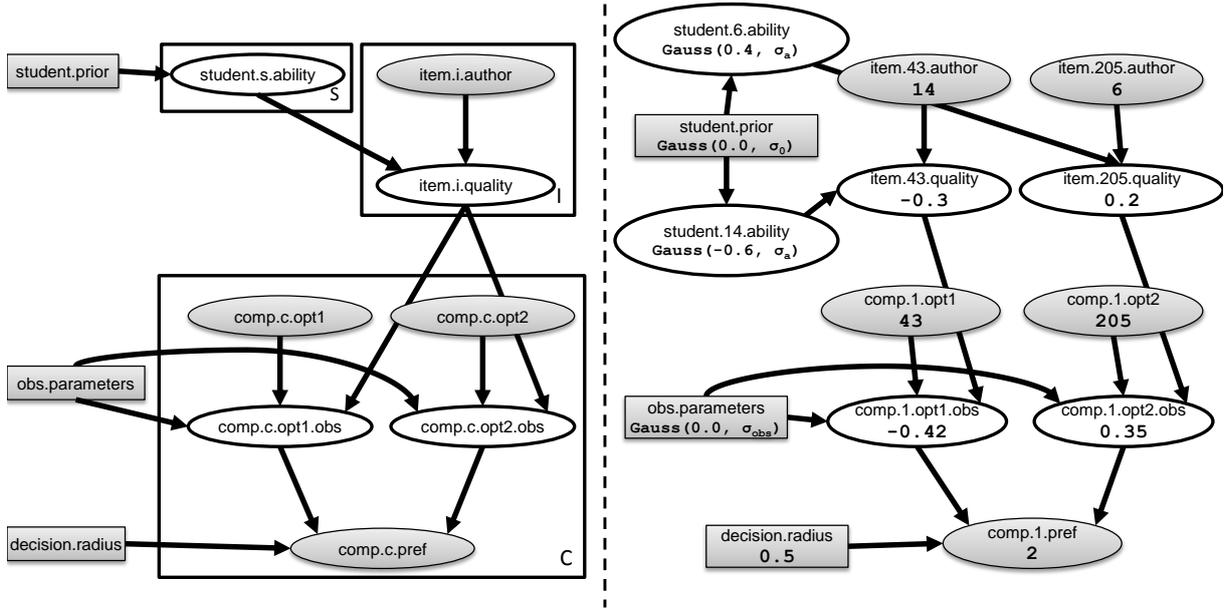


Figure 1: Plate notation (left) showing the independence structure of the IRT Models. Example instantiated subnetwork (right) for the Gaussian parameterization. Shaded rectangles are hyperparameters. Shaded ellipses are variables observable from a set of comparisons.

radius (which here is 0.5), then he states no preference (i.e. a preference of 0). Otherwise he prefers the item with the higher observed quality.

The categorical parameterization is similar to the Gaussian parameterization, with the following differences. Item quality is not continuous, but rather a member of the discrete set  $\{1, 2, \dots, \Lambda\}$ . The student ability distributions are categorical distributions over  $\{1, 2, \dots, \Lambda\}$ , and the student ability prior is a symmetric Dirichlet with strength  $\alpha_a$ . Finally, the observed quality is the item quality  $\lambda$  plus an integer-valued noise  $\nu \in \{1 - \lambda, \dots, \Lambda - \lambda\}$ . Noise  $\nu$  is drawn from a discretized zero-mean Gaussian with standard deviation  $\sigma_{obs}$ . Specifically,  $Pr(\nu)$  is proportional to the value of the probability density function of the zero-mean Gaussian  $\mathcal{N}(0, \sigma_{obs})$ .

We estimated the model parameters with Gibbs sampling (Geman and Geman, 1984). We found that Gibbs sampling converged quickly and consistently<sup>10</sup> for both parameterizations. Given the parameter estimates, we obtain a preference model  $Q(\pi|s_1, s_2)$  through the inference query:

$$Pr(\text{comp.c}'.\text{pref} = \pi \mid \begin{array}{l} \text{item.i}'.\text{author} = s_1, \\ \text{item.i}''.\text{author} = s_2, \\ \text{comp.c}'.\text{opt1} = i', \\ \text{comp.c}'.\text{opt2} = i'' \end{array})$$

<sup>10</sup>We ran 200 iterations with a burn-in of 50.

where  $c', i', i''$  are new comparison and item ids that do not appear in the training data.

We call these models Item-Response Theoretic (IRT) models, to acknowledge their roots in the psychometrics (Thurstone, 1927; Bradley and Terry, 1952; Luce, 1959) and item-response theory (Hambleton, 1991; van der Linden and Hambleton, 1996; Baker, 2001) literature. Item-response theory is the basis of modern testing theory and drives adaptive standardized tests like the Graduate Record Exam (GRE). In particular, the Gaussian parameterization of our IRT models strongly resembles<sup>11</sup> the Thurstone (Thurstone, 1927) and Bradley-Terry-Luce (Bradley and Terry, 1952; Luce, 1959) models of paired comparison and the 1PL normal-ogive and Rasch (Rasch, 1960) models of student testing. From the testing perspective, we can view each comparison as two students simultaneously posing a test question to the other: “Give me a translation of the source text which is better than mine.” The students can answer the question correctly, incorrectly, or they can provide a translation of analogous quality. An extra dimension of our models is judge noise, not a factor when modeling multiple-choice tests, for which the right answer is not subject to opinion.

<sup>11</sup>These models are not traditionally expressed using graphical models, although it is not unprecedented (Mislevy and Almond, 1997; Mislevy et al., 1999).

lp	wmt10		wmt11		wmt12	
	train	test	train	test	train	test
ce	3166	2209	1706	3216	5969	6806
fe	5918	2376	2556	4430	7982	5840
ge	7422	3002	3708	5371	8106	6032
se	8411	2896	1968	3684	3910	7376
ec	10490	3048	8859	9016	13770	9112
ef	5720	2242	3328	5758	7841	7508
eg	10852	2842	5964	7032	10210	7191
es	2962	2212	4768	6362	5664	8928

Table 3: Dataset sizes for each competition track (number of comparisons).

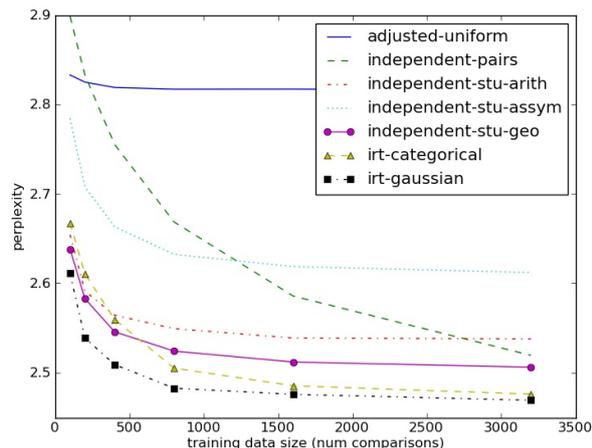


Figure 2: WMT10 model perplexities. The perplexity of the uniform preference model is 3.0 for all training sizes.

## 8 Experiments

We organized the competition data as described at the end of Section 4. To compare the preference models, we did the following:

- Randomly chose a subset of  $k$  comparisons from the training set, for  $k \in \{100, 200, 400, 800, 1600, 3200\}$ .<sup>12</sup>
- Trained the preference model on these comparisons.
- Evaluated the perplexity of the trained model on the test preferences, as described in Section 4.

For each model and training size, we averaged the perplexities from 5 trials of each competition track. We then plotted average perplexity as a function of training size. These graphs are shown

<sup>12</sup>If  $k$  was greater than the total number of training comparisons, then we took the entire set.

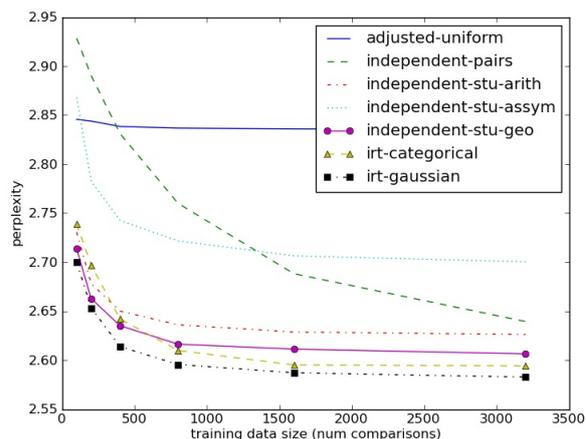


Figure 3: WMT11 model perplexities.

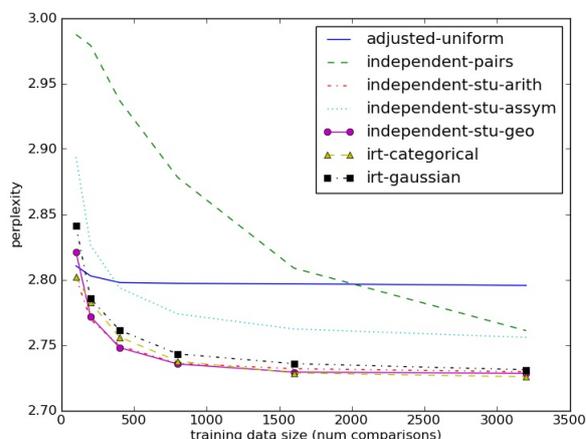


Figure 4: WMT12 model perplexities.

in Figure 2 (WMT10)<sup>13</sup>, and Figure 4 (WMT12). For WMT10 and WMT11, the best models were the IRT models, with the Gaussian parameterization converging the most rapidly and reaching the lowest perplexity. For WMT12, in which reference translations were excluded from the competition, four models were nearly indistinguishable: the two IRT models and the two averaged Independent Student models. This somewhat validates the organizers' decision to exclude the references, particularly given WMT's use of the BOJAR ranking heuristic (the nucleus of the Independent Student models) for its official rankings.

<sup>13</sup>Results for WMT10 exclude the German-English and English-German tracks, since we used these to tune our model hyperparameters. These were set as follows. The Dirichlet strength for each baseline was 1. For IRT-Gaussian:  $\sigma_0 = 1.0, \sigma_{obs} = 1.0, \sigma_a = 0.5$ , and the decision radius was 0.4. For IRT-Categorical:  $\Lambda = 8, \sigma_{obs} = 1.0, \alpha_a = 0.5$ , and the decision radius was 0.

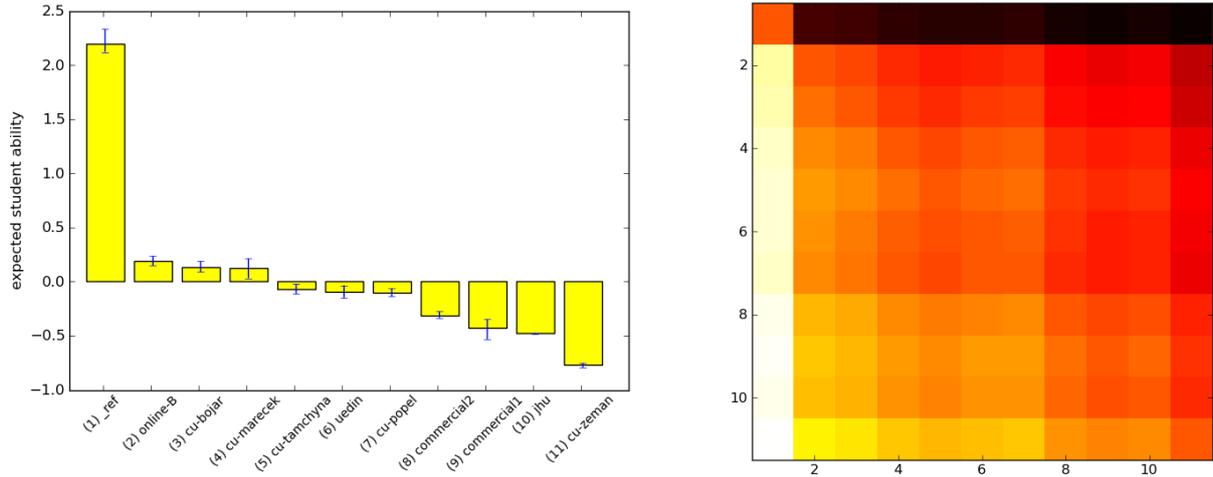


Figure 6: English-Czech WMT11 results (average of 5 trainings on 1600 comparisons). Error bars (left) indicate one stddev of the estimated ability means. In the heatmap (right), cell  $(s_1, s_2)$  is darker if preference model  $Q(\pi|s_1, s_2)$  skews in favor of student  $s_1$ , lighter if it skews in favor of student  $s_2$ .

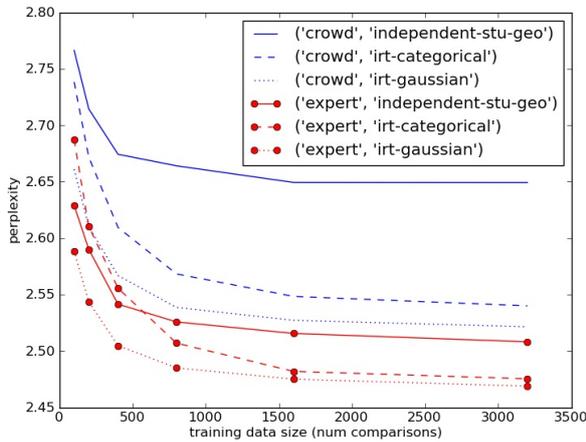


Figure 5: WMT10 model perplexities (crowd-sourced versus expert training).

The IRT models proved the most robust at handling judge noise. We repeated the WMT10 experiment using the same test sets, but using the unfiltered crowdsourced comparisons (rather than “expert”<sup>14</sup> comparisons) for training. Figure 5 shows the results. Whereas the crowdsourced noise considerably degraded the Geometric Independent Students model, the IRT models were remarkably robust. IRT-Gaussian in particular came close to replicating the performance of Geometric Independent Students trained on the much cleaner expert data. This is rather impressive, since the crowdsourced judges agree only 46.6% of the time, compared to a 65.8% agreement rate among

<sup>14</sup>I.e., machine translation specialists.

expert judges (Callison-Burch et al., 2010).

Another nice property of the IRT models is that they explicitly model student ability, so they yield a natural ranking. For training size 1600 of the WMT11 English-Czech track, Figure 6 (left) shows the mean student abilities learned by the IRT-Gaussian model. The error bars show one standard deviation of the ability means (recall that we performed 5 trials, each with a random training subset of size 1600). These results provide further insight into a case analyzed by (Lopez, 2012), which raised concern about the relative ordering of online-B, cu-bojar, and cu-marecek. According to IRT-Gaussian’s analysis of the data, these three students are so close in ability that any ordering is essentially arbitrary. Short of a full ranking, the analysis does suggest four strata. Viewing one of IRT-Gaussian’s induced preference models as a heatmap<sup>15</sup> (Figure 6, right), four bands are discernable. First, the reference sentences are clearly the darkest (best). Next come students 2-7, followed by the slightly lighter (weaker) students 8-10, followed by the lightest (weakest) student 11.

## 9 Conclusion

WMT has faced a crisis of confidence lately, with researchers raising (real and conjectured) issues with its analytical methodology. In this paper, we showed how WMT can restore confidence in

<sup>15</sup>In the heatmap, cell  $(s_1, s_2)$  is darker if preference model  $Q(\pi|s_1, s_2)$  skews in favor of student  $s_1$ , lighter if it skews in favor of student  $s_2$ .

its conclusions – by shifting the focus from *rankings* to *relative ability*. Estimates of relative ability (the expected head-to-head performance of system pairs over a probability space of judges and source text) can be empirically compared, granting substance to previously nebulous questions like:

1. **Is my analysis better than your analysis?**  
Rather than the current anecdotal approach to comparing competition analyses (e.g. presenting example rankings that seem somehow wrong), we can empirically compare the predictive power of the models on test data.
2. **How much of an impact does judge noise have on my conclusions?** We showed that judge noise can have a significant impact on the quality of our conclusions, if we use the wrong models. However, the IRT-Gaussian appears to be quite noise-tolerant, giving similar-quality conclusions on both expert and crowdsourced comparisons.
3. **How many comparisons should I elicit?** Many of our preference models (including IRT-Gaussian and Geometric Independent Students) are close to convergence at around 1000 comparisons. This suggests that we can elicit far fewer comparisons and still derive confident conclusions. This is the first time a concrete answer to this question has been provided.

## References

- F.B. Baker. 2001. *The basics of item response theory*. ERIC.
- Ondej Bojar, Miloš Ercegović, Martin Popel, and Omar Zaidan. 2011. A grain of salt for the wmt manual evaluation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 1–11, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.
- C. Callison-Burch, P. Koehn, C. Monz, K. Peterson, M. Przybocki, and O.F. Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53. Association for Computational Linguistics.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*.
- S. Geman and D. Geman. 1984. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741.
- R.K. Hambleton. 1991. *Fundamentals of item response theory*, volume 2. Sage Publications, Incorporated.
- D. Koller and N. Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.
- S. Kullback and R.A. Leibler. 1951. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.
- Adam Lopez. 2012. Putting human assessments of machine translation systems in order. In *Proceedings of WMT*.
- R. Duncan Luce. 1959. *Individual Choice Behavior a Theoretical Analysis*. John Wiley and sons.
- R.J. Mislevy and R.G. Almond. 1997. Graphical models and computerized adaptive testing. *UCLA CSE Technical Report 434*.
- R.J. Mislevy, R.G. Almond, D. Yan, and L.S. Steinberg. 1999. Bayes nets in educational assessment: Where the numbers come from. In *Proceedings of the fifteenth conference on uncertainty in artificial intelligence*, pages 437–446. Morgan Kaufmann Publishers Inc.
- G. Rasch. 1960. Studies in mathematical psychology: I. probabilistic models for some intelligence and attainment tests.
- Louis L Thurstone. 1927. A law of comparative judgment. *Psychological review*, 34(4):273–286.
- W.J. van der Linden and R.K. Hambleton. 1996. *Handbook of modern item response theory*. Springer.

# Learning a Phrase-based Translation Model from Monolingual Data with Application to Domain Adaptation

Jiajun Zhang and Chengqing Zong

National Laboratory of Pattern Recognition

Institute of Automation, Chinese Academy of Sciences, Beijing, China

{jjzhang, cqzong}@nlpr.ia.ac.cn

## Abstract

Currently, almost all of the statistical machine translation (SMT) models are trained with the parallel corpora in some specific domains. However, when it comes to a language pair or a different domain without any bilingual resources, the traditional SMT loses its power. Recently, some research works study the unsupervised SMT for inducing a simple word-based translation model from the monolingual corpora. It successfully bypasses the constraint of bitext for SMT and obtains a relatively promising result. In this paper, we take a step forward and propose a simple but effective method to induce a phrase-based model from the monolingual corpora given an automatically-induced translation lexicon or a manually-edited translation dictionary. We apply our method for the domain adaptation task and the extensive experiments show that our proposed method can substantially improve the translation quality.

## 1 Introduction

During the last decade, statistical machine translation has made great progress. Novel translation models, such as phrase-based models (Koehn et al., 2007), hierarchical phrase-based models (Chiang, 2007) and linguistically syntax-based models (Liu et al., 2006; Huang et al., 2006; Galley, 2006; Zhang et al., 2008; Chiang, 2010; Zhang et al., 2011; Zhai et al., 2011, 2012) have been proposed and achieved higher and higher translation performance. However, all of these state-of-the-art translation models rely on the parallel corpora to induce translation rules and estimate the corresponding parameters.

It is unfortunate that the parallel corpora are very expensive to collect and are usually not available for resource-poor languages and for many specific domains even in a resource-rich language pair.

Recently, more and more researchers concentrated on taking full advantage of the monolingual corpora in both source and target languages, and proposed methods for bilingual lexicon induction from non-parallel data (Rapp, 1995, 1999; Koehn and Knight, 2002; Haghighi et al., 2008; Daumé III and Jagarlamudi, 2011) and proposed unsupervised statistical machine translation (bilingual lexicon is a byproduct) with only monolingual corpora (Ravi and Knight, 2011; Nuhn et al., 2012; Dou and Knight, 2012).

In the bilingual lexicon induction (Koehn and Knight, 2002; Haghighi et al., 2008; Daumé III and Jagarlamudi, 2011), with the help of the orthographic and context features, researchers adopted an unsupervised method, such as canonical correlation analysis (CCA) model, to automatically induce the word translation pairs between two languages from non-parallel data only requiring that the monolingual data in each language are from a fairly comparable domain.

The unsupervised statistical machine translation method (Ravi and Knight, 2011; Nuhn et al., 2012; Dou and Knight, 2012) viewed the translation task as a decipherment problem and designed a generative model with the objective function to maximize the likelihood of the source language monolingual data. To tackle the large-scale vocabulary, they mainly considered the word-based model (e.g. IBM Model 3) and applied the Bayesian method with Gibbs sampling or slice sampling. Finally, they used the learned translation model directly to translate unseen data (Ravi and Knight, 2011; Nuhn et al., 2012) or incorporated the learned bilingual lexicon as a new in-domain translation resource into the phrase-based model which is trained with out-of-domain data to improve the domain adaptation performance in machine translation (Dou and Knight, 2012).

We can easily see that these unsupervised methods can only induce the word-based translation rules (bilingual lexicon) at present. It is a big challenge that whether we can induce phrase

---

1, word reordering example:
本 发明 的 目的 在于     the purpose of the invention is to     0-0 0-3 1-4 2-2 3-1 4-5 4-6
2, idiom example:
辨识 <i>真伪</i> 的     distinguish the <i>true</i> from the <i>false</i>     0-0 1-2 1-5 2-1 2-4
3, unknown word translation:
发光 <i>二极管</i> 芯片 的     of the light-emitting <i>diode</i> chip     0-2 1-2 2-4 3-0 3-1

---

Table 1: Examples of new translation knowledge learned with the proposed phrase pair induction method. For the three fields separated by “|||”, the first two are respectively Chinese and English phrase, and the last one is the word alignment between these two phrases.

level translation rules and learn a phrase-based model from the monolingual corpora.

In this paper, we focus on exploring this direction and propose a simple but effective method to induce the phrase-level translation rules from monolingual data. The main idea of our method is to divide the phrase-level translation rule induction into two steps: bilingual lexicon induction and phrase pair induction.

Since many researchers have studied the bilingual lexicon induction, in this paper, we mainly concentrate ourselves on phrase pair induction given a probabilistic bilingual lexicon and two in-domain large monolingual data (source and target language). In addition, we will further introduce how to refine the induced phrase pairs and estimate the parameters of the induced phrase pairs, such as four standard translation features and phrase reordering feature used in the conventional phrase-based models (Koehn et al., 2007). The induced phrase-based model will be used to help domain adaptation for machine translation.

In the rest of this paper, we first explain with examples to show what new translation knowledge can be learned with our proposed phrase pair induction method (Section 2), and then we introduce the approach for probabilistic bilingual lexicon acquisition in Section 3. In Section 4 and 5, we respectively present our method for phrase pair induction and introduce an approach for phrase pair refinement and parameter estimation. Section 6 will show the detailed experiments for the task of domain adaptation. We will introduce some related work in Section 7 and conclude this paper in Section 8.

## 2 What Can We Learn with Phrase Pair Induction?

Readers may doubt that if phrase pair induction is performed only using bilingual lexicon and monolingual data, what new translation knowledge can be learned?

The bilingual lexicon can only express the translation equivalence between source- and target-side word pair and has little ability to deal with word reordering and idiom translation. In contrast, phrase pair induction can make up for this deficiency to some extent. Furthermore, our method is able to learn some unknown word translations.

From the induced phrase pairs with our method, we have conducted a deep analysis and find that we can learn three kinds of new translation knowledge: 1) word reordering in a phrase pair; 2) idioms; and 3) unknown word translations. Table 1 gives examples for each of the three kinds. For the first example, the source and target phrase are extracted respectively from monolingual data, each word in the source phrase has a translation in the target phrase, but the word order is different. The word order encoded in a phrase pair is difficult to learn in a word-based SMT. In the second example, the *italic* source word corresponds to two target words (in *italic*), and the phrase pair is an idiom which cannot be learned from word-based SMT. In the third example, as we learn from the source and target monolingual text that the words around the *italic* ones are translations with each other, thus we cannot only extract a new phrase pair but also learn a translation pair of unknown words in *italic*.

## 3 Probabilistic Bilingual Lexicon Acquisition

In order to induce the phrase pairs from the in-domain monolingual data for domain adaptation, the probabilistic bilingual lexicon is essential.

In this paper, we acquire the probabilistic bilingual lexicon from two approaches: 1) build a bilingual lexicon from large-scale out-of-domain parallel data; 2) adopt a manually collected in-domain lexicon. This paper uses Chinese-to-English translation as a case study and electronic data is the in-domain data we focus on.

In Chinese-to-English translation, there are lots of parallel data on News. Here, we utilize about 2.08 million sentence pairs<sup>1</sup> in News domain to learn a probabilistic bilingual lexicon. Basically, we can use GIZA++ (Och, 2003) to get the probabilistic lexicon. However, the problem is that each source-side word associates too many possible translations which contain much noise. For instance, in the lexicon obtained with GIZA++, each source-side word has about 13 translations on average. The noise of the lexicon can influence the accuracy of the induced phrase pairs to a large extent. To learn a lexicon with a high precision, we follow Munteanu and Marcu (2006) to apply *Log-Likelihood-Ratios* (Dunning, 1993; Melamed, 2000; Moore, 2004a, 2004b) to estimate how strong the association is between a source-side word and its aligned target-side word. We employ the same algorithm used in (Munteanu and Marcu, 2006) which first use the GIZA++ (with grow-diag-final-and heuristic) to obtain the word alignment between source and target words, and then calculate the association strength between the aligned words. After using the log-likelihood-ratios algorithm<sup>2</sup>, we obtain a probabilistic bilingual lexicon with bidirectional translation probabilities from the out-of-domain data. In the final lexicon, the number of average translations is only 5. We call this lexicon *LLR-lex*.

In the electronic domain, we manually collected a lexicon which contains about 140k entries. It should be noted that there is no translation probability in this lexicon. In order to assign probabilities to each entry, we apply the *Corpus Translation Probability* which used in (Wu et al., 2008): given an in-domain source language monolingual data, we translate this data with the phrase-based model trained on the out-of-domain News data, the in-domain lexicon and the in-domain target language monolingual data (for language model estimation). With the source language data and its translation, we estimate the bidirectional translation probabilities for each entry in the original lexicon. For the entries whose translation probabilities are not estimated, we just assign a uniform probability. That is if a source word has  $n$  translations, then the translation probability of target word given the source word is  $1/n$ . We call this lexicon *Domain-lex*.

<sup>1</sup> LDC category numbers are: LDC2000T50, LDC2003E14, LDC2003E07, LDC2004T07, LDC2005T06, LDC2002L27, LDC2005T10 and LDC2005T34.

<sup>2</sup> Following Moore (2004b), we use the threshold 10 on LLR to filter out unlikely translations.

We combine *LLR-lex* and *Domain-lex* to obtain the final probabilistic bilingual lexicon for phrase pair induction.

## 4 Phrase Pair Induction Method

Given a probabilistic bilingual lexicon and two monolingual data, we present a simple but effective method for phrase pair induction in this section.

---

Input: Probabilistic bilingual lexicon  $V$  (each source word  $s$  maps a translation set  $V[s]$ )  
 Source language monolingual data  $S=\{s_n\} n=1\dots N$   
 Target language monolingual data  $T=\{t_m\} m=1\dots M$   
 Output: Phrase pairs  $P$

---

```

1: For each distinct source-side phrase  $s_i^j$  in  $S$ :
2:   If each  $s_k \in s_i^j$  in  $V$ :
3:     Collect  $V[s_k]_{k=i}^j$ 
4:     For each permutation  $s_i^{j'}$  of  $s_i^j$ :
5:       If  $t_{i'}^{j'}$  in  $T$ :  $\triangleright t_{k'} \in V[s_{k'}] k' \in [i, j]$ 
6:         Add phrase pair  $(s_i^j, t_{i'}^{j'})$  into  $P$ 

```

---

Figure 1: a naïve algorithm for phrase pair induction.

### 4.1 A Naïve Method

We first introduce a relatively naïve way to extract phrase pairs from the given resources. For a source phrase (word sequence), we can reorder the words in the phrase (permutation) first, and then obtain the target phrases with the bilingual lexicon (translation), and finally check if the target phrase is in the target monolingual data. The algorithm is given in Figure 1.

Figure 1 shows that the naïve algorithm is very easy to implement. However, the time complexity is too high. For each source phrase  $s_i^j$  (with  $(j-i+1)!$  permutations), suppose a source word has  $C$  translations on average and checking whether the target phrase  $t_{i'}^{j'}$  in  $T$  needs time  $O(|T|)$ , then, phrase pair induction for a single source phrase needs time  $O(C^{j-i+1}|T|(j-i+1)!)$ .

It is very time consuming. One may design smarter algorithms. For example, one can collect distinct  $n$ -grams from source and target monolingual data. Then, for a source-side phrase with length  $L$ , one can find the best translation candidate using the probabilistic bilingual lexicon from the target-side phrases with the same length  $L$ . The biggest disadvantage of these algorithms is that they can only induce phrase pair (with the

same length) encoding word reordering, but cannot learn phrase pairs in different length. Furthermore, they cannot learn idioms and unknown word translations from monolingual data. Obviously, these kind of approaches is not optimal.

## 4.2 Phrase Pair Induction with Inverted Index

In order to make the phrase pair induction both effective and efficient, we propose a method using inverted index data structure which is usually a central component of a typical search engine.

The inverted index is employed to represent the target language monolingual data. For a target language word, the inverted index not only records the sentence position in monolingual data, but also records the word position in a sentence. Some examples are shown in Table 2. By doing this, we do not need to iterate all the permutations of source language phrase  $s_i^j$  to explore possible phrase pairs encoding word reordering. Furthermore, it is possible to learn idiom translation and unknown word translations. We will elaborate how to induce phrase pairs with the help of inverted index.

Target Language Word	Position
communication	(2,5), (106,20), ..., (23022, 12)
...	...
zoom	(90,2), (280,21), ..., (90239,15)

Table 2: Some examples of inverted index for target language words, (2,5) means that “communication” occurs at the 5<sup>th</sup> word of the 2<sup>nd</sup> sentence in the target monolingual data.

The new algorithm for phrase pair induction is presented in Figure 2. Line 1 iterates all the distinct phrases in the source-side monolingual data. It can be implemented by collecting all the distinct n-grams in which n is the phrase length we are interested in (3 to 7 in this paper). For each distinct source-side phrase, Line 2-5 efficiently collects all the positions in the target monolingual data for the translations of each word in the source phrase. Line 6 sorts the positions so that we can easily find the position sequence belonging to a same sentence. Line 8-9 discards all the position sub-sequences that lack translations for more than one source-side words. That is to say we allow at most one unknown word in an induced phrase pair in order to make the induction more accurate. Line 10 and Line 12 is the core of this algorithm. We first define a constraint before detailing the algorithm.

---

Input: Probabilistic bilingual lexicon  $V$  (each source word  $s$  maps a translation set  $V[s]$ )  
Source language monolingual data  $S=\{s_n\} n=1\dots N$   
Inverted index representing target language monolingual data  $IMap$

---

Output: Phrase pairs  $P$

---

```

1: For each distinct source-side phrase  $s_i^j$  in  $S$ :
2:    $positionArray = []$ 
3:   For each  $s_k \in s_i^j$ :
4:     For each  $t \in V[s_k]$ :
5:       add  $IMap[t]$  into  $positionArray$ 
6:   Sort  $positionArray$ 
7:   For each sequence in a same sentence in  $positionArray$ :
8:     If more than 1 word in  $s_i^j$  has no trans in the seq:
9:       Discard this seq and continue
10:    Probability smoothing for single word gap
11:    For all continuous position sub-sequence:
12:      Find the one  $t_h^k$  with maximum probability
13:    Add phrase pair  $(s_i^j, t_h^k)$  into  $P$ 

```

---

Figure 2: Phrase pair induction using inverted index.

**Constraint:** we require that there exists at most one phrase in a target sentence that is the translation of the source-side phrase.

According to our analysis, it is not often to find that two phrases (length larger than 2) in a same sentence have the same meaning. Even if it happens, it is reasonable to keep the one with the highest probability. Given a position sequence belonging to a same sentence, Line 10 smoothes the probability of the single word gap according to the probabilities of the around words. Single word gap means that this word is not aligned but its left and right words are aligned with the words of the source-side phrase. Suppose the target sub-sequence is  $t_i \dots t_{i+r} \dots t_j$  and  $t_{i+r}$  is the only word that is not aligned with source-side words. We smooth the probability  $p(t_{i+r} | null)$  as follows:

$$p(t_{i+r} | null) = \begin{cases} \frac{\min\{p(t_i | s_i), p(t_j | s_j)\}}{2}, & \text{if } r=1 \text{ or } r+1=j \\ \frac{p(t_{i+r-1} | s_{i+r-1}) + p(t_{i+r+1} | s_{i+r+1})}{2}, & \text{otherwise} \end{cases} \quad (1)$$

The above formula means that if the left or the right side only has one word, then the smoothed probability is one half of the minimum of the probabilities of the two neighbors, otherwise the smoothed probability is the average of the probabilities of the two neighbors. This smoothing strategy encourages that if more words around the un-aligned word are translations of the source-side phrase, then the gap word is more likely to belong to the translations of the source-side phrase.

After probability smoothing of the single gap word, we are ready to extract the candidate translation of the source-side phrase. Similar with Line 9 in Figure 2, we further filter the target continuous phrase if more than one word in source-side phrase has no translation in this target phrase. After that, we just choose the continuous target phrase with the largest probability if two or more continuous target phrases exist in the same target sentence. The probability of a target-side phrase given the source-side phrase is computed similar to that of (Koehn et al., 2003) except that we impose length normalization:

$$p_{lex}(t|s,a) = \left[ \prod_{i=1}^n \frac{1}{|\{j|(i,j) \in a\}|} \sum_{\forall(i,j) \in a} p(t_i|s_j) \right]^{\frac{1}{n}} \quad (2)$$

where the alignment  $a$  is produced using probabilistic bilingual lexicon. If a target word in  $t$  is a gap word, we suppose there is a word alignment between the target gap word and the source-side *null*.

Similarly, we can compute the probability of source-side phrase given the target-side phrase  $p_{lex}(s|t,a)$ . Then, we find the target-side phrase which has the biggest value of  $p_{lex}(t|s,a) \cdot p_{lex}(s|t,a)$ . Line 13 in Figure 2 collects the induced phrase pairs.

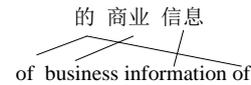
For the time complexity, it depends on the length of *positionArray*, since the time complexity of the core algorithm (Line 7-13) is proportional to the length of *positionArray*. If *positionArray* contains almost all the positions in the target monolingual data  $T$ , then the worst time complexity will be  $O(|T|\log|T|)$  (for array sort). However, we find in the target monolingual data (1 million sentences) that each distinct word happens 110 times on average. Then, for a source-side phrase with 7 words, the average length of *positionArray* will be 3850, since each source word has averagely 5 target translations (mentioned in Section 3). Therefore, the algorithm is relatively efficient in the average case.

## 5 Phrase Pair Refinement and Parameterization

### 5.1 Phrase Pair Refinement

Some of the phrase pairs induced in Section 4 may contain noise. According to our analysis, we find that the biggest problem is that in the target-side of the phrase pair, there are two or more identical words aligned to the same source-

side word. For example, we extract a phrase pair as follows:



In the above phrase pair, there are two words “of” in the target side and the first one is redundant. The phrase pair induction algorithm presented in Section 4 cannot deal with this situation. In this section, we propose a simple approach to handle this problem. For each entry in *LLR-lex*, such as (的, of), we can learn two kinds of information from the out-of-domain word-aligned sentence pairs: one is whether the target translation is before or after the translation of the preceding source-side word (Order); the other is whether the target translation is adjacent with the translation of the preceding source-side word (Adjacency). If the source-side word is the beginning of the phrase, we calculate the corresponding information with the succeeding word instead of the preceding word. For the entries in *Domain-lex*, we constrain that the target translation should be adjacent with the translations of its source-side neighbors and translation order is the same with the source-side words.

With the Order and Adjacency information, we first check the order information, and then check the adjacency information if the duplicates cannot be handled using order information. For example, since (的, of) is an entry in *LLR-lex* and we have learned that “of” is much more likely to be behind the translation of the succeeding word. Thus, the first word “of” can be discarded. This refinement can be applied before finding the phrase pair with maximum probability (Line 12 in Figure 2) so that the duplicate words do not affect the calculation of translation probability of phrase pair.

### 5.2 Translation Probability Estimation

It is well known that in the phrase-based SMT there are four translation probabilities and the reordering probability for each phrase pair.

The translation probabilities in the traditional phrase-based SMT include bidirectional phrase translation probabilities and bidirectional lexical weights. For the lexical weights, we can use the  $p_{lex}(s|t,a)$  and  $p_{lex}(t|s,a)$  computed in the above section without length normalization. However, for the phrase-level probability, we cannot use maximum likelihood estimation since the phrase pairs are not extracted from parallel sentences.

In this paper, we borrow and extend the idea of (Klementiev et al., 2012) to calculate the phrase-level translation probability with context information in source and target monolingual corpus. The value is calculated using a vector space model. With source and target vocabularies  $(s_1, s_2, \dots, s_N)$  and  $(t_1, t_2, \dots, t_M)$ , the source-side phrase  $s$  and target-side phrase  $t$  can be respectively represented in an  $N$ - and  $M$ -dimensional vector. The  $k$ -th component of  $s$ 's contextual vector is computed using the method of (Fung and Yee, 1998) as follows:

$$w_k = n_{s,k} \times (\log(n_{\max} / n_k) + 1) \quad (3)$$

where  $n_{s,k}$  and  $n_k$  denotes the number of times  $s_k$  occurs in the context of  $s$  and in the entire source language monolingual data, and  $n_{\max}$  is the maximum number of occurrence of any source-side word in the source language monolingual data. The  $k$ -th element of  $t$ 's vector can be computed with the same method. We finally normalize these vectors with  $L_2$ -norm.

With the  $s$ 's and  $t$ 's contextual vector representations, we calculate two similarities: 1) project  $s$ 's vector into target side  $\hat{t}$  with the lexical mapping  $p(t/s)$ , and then get the similarity by computing the cosine of two angles between  $t$ 's and  $\hat{t}$ 's vectors; 2) project  $t$ 's vector into source side  $\hat{s}$  with the lexical mapping  $p(s/t)$ , and then obtain the similarity between  $s$ 's and  $\hat{s}$ 's vectors. These two contextual similarities will serve as two phrase-level translation probabilities.

### 5.3 Reordering Probability Estimation

For the reordering probabilities of newly induced phrase pairs, we can also follow Klementiev et al. (2012) to estimate these probabilities using source and target monolingual data. The method is to calculate six probabilities for monotone, swap or discontinuous cases. For the phrase pair (的商业信息, business information of), we find a source sentence containing 的商业信息, and find a target sentence containing *business information of*. If there is another phrase pair  $(\bar{s}, \bar{t})$ ,  $\bar{t}$  exactly follows *business information of* and  $\bar{s}$  occurs in the same source sentence with 的商业信息, then we compare the position relationship between  $\bar{s}$  and 的商业信息. We increment the swap count if  $\bar{s}$  is just before 的商业信息. After counting, we finally use maximum likelihood estimation method to compute the reordering probabilities.

## 6 Related Work

As far as we know, few researchers study phrase pair induction from only monolingual data.

There are three research works that are most related with ours. One is using an in-domain probabilistic bilingual lexicon to extract sub-sentential parallel fragments from comparable corpora (Munteanu and Marcu, 2006; Quirk et al., 2007; Cettolo et al., 2010). Munteanu and Marcu (2006) first extract the candidate parallel sentences from the comparable corpora and further extract the accurate sub-sentential bilingual fragments from the candidate parallel sentences using the in-domain probabilistic bilingual lexicon. Compared with their work, our focus is to induce phrase pairs directly from monolingual data rather than comparable data. Thus, finding the candidate parallel sentences is not possible in our situation.

Another is to make full use of monolingual data with transductive learning (Ueffing et al., 2007; Schwenk, 2008; Wu et al., 2008; Bertoldi and Federico, 2009). For the target-side monolingual data, they just use it to train language model, and for the source-side monolingual data, they employ a baseline (word-based SMT or phrase-based SMT trained with small-scale bitext) to first translate the source sentences, combining the source sentence and its target translation as a bilingual sentence pair, and then train a new phrase-base SMT with these pseudo sentence pairs. This method cannot learn idiom translations and unknown word translations.

The third is to estimate the translation parameters and reordering parameters using monolingual data given the phrase pairs (Klementiev et al., 2012). Their work supposes the phrase pairs are already given and then corresponding parameters can be learned with monolingual data. Different from their work, we concentrate ourselves on inducing phrase pairs from monolingual data and then borrow some ideas from theirs for parameter estimation. Furthermore, we extend their contextual similarity between source and target phrases to both directions.

## 7 Experiments

### 7.1 Experimental Setup

Our purpose is to induce phrase pairs to improve translation quality for domain adaptation. We have introduced the out-of-domain data and the electronic in-domain lexicon in Section 3. Here we introduce other information about the in-

domain data. Besides the in-domain lexicon, we have collected respectively 1 million monolingual sentences in electronic area from the web. They are neither parallel nor comparable because we cannot even extract a small number of parallel sentence pairs from this monolingual data using the method of (Munteanu and Marcu, 2006). We further employ experts to translate 2000 Chinese electronic sentences into English. The first half is used as the tuning set (*elec1000-tune*) and the second half is employed as the testing set (*elec1000-test*).

We construct two kinds of phrase-based models using Moses (Koehn et al., 2007): one uses out-of-domain data and the other uses in-domain data. For the out-of-domain data, we build the phrase table and reordering table using the 2.08 million Chinese-to-English sentence pairs, and we use the SRILM toolkit (Stolcke, 2002) to train the 5-gram English language model with the target part of the parallel sentences and the Xinhua portion of the English Gigaword. For the in-domain electronic data, we first consider the lexicon as a phrase table in which we assign a constant 1.0 for each of the four probabilities, and then we combine this initial phrase table and the induced phrase pairs to form the new phrase table. The in-domain reordering table is created for the induced phrase pairs. An in-domain 5-gram English language model is trained with the target 1 million monolingual data.

We use BLEU (Papineni et al., 2002) score with shortest length penalty as the evaluation metric and apply the pairwise re-sampling approach (Koehn, 2004) to perform the significance test.

## 7.2 Experimental Results

In this section, we first conduct experiments to figure out how the translation performance degrades when the domain changes. To better illustrate the comparison, we first use News data to evaluate the NIST evaluation tests and then use the same News data to evaluate the electronic test sets. For the NIST evaluation, we employ Chinese-to-English NIST MT03 as the tuning set and NIST MT05 as the test set. Table 3 gives the results. It is obvious that, it is relatively high when using the News training data to evaluate the same News test set. However, when the test domain is changed, the translation performance decreases to a large extent.

Given the in-domain bilingual lexicon and two monolingual data, previous works also proposed

some good methods to explore the potential of the given data to improve the translation quality. Here, we implement their approaches and use them as our strong baseline. Wu et al. (2008) regards the in-domain lexicon with corpus translation probability as another phrase table and further use the in-domain language model besides the out-of-domain language model. Table 4 gives the results. We can see from the table that the domain lexicon is much helpful and significantly outperforms the baseline with more than 4.0 BLEU points. When it is enhanced with the in-domain language model, it can further improve the translation performance by more than 2.5 BLEU points. This method has made good use of in-domain lexicon and the target-side in-domain monolingual data, but it does not take full advantage of the in-domain source-side monolingual data.

In order to use source-side monolingual data, Ueffing et al. (2007), Schwenk (2008), Wu et al. (2008) and Bertoldi and Federico (2009) employed the transductive learning to first translate the source-side monolingual data using the best configuration (baseline+in-domain lexicon+in-domain language model) and obtain 1-best translation for each source-side sentence. With the source-side sentences and their translations, the new phrase table and reordering table are built. Then, these resources are added into the best configuration. The experimental results are presented in the last row of Table 4. From the results, we see that transductive learning can further improve the translation performance significantly by 0.6 BLEU points.

In transductive learning, in-domain lexicon and both-side monolingual data have been explored. However, this method does not take full advantage of both-side monolingual data because it uses source and target monolingual data individually. In our method, we explore fully the source and target monolingual data to induce translation equivalence on the phrase level. In order to make the phrase pair induction more efficient, we first sort all the sentences in the both-side monolingual data according to the word hit rate in the bilingual lexicon. Then, we conduct six sets of experiments respectively on the first 100k, 200k, 300k, 500k and whole 1m sentences. All the experiments are run based on the configuration with BLEU 13.41 in Table 4, and we call this configuration *BestConfig*. Note that the unknown words are only allowed if the source-side of a phrase pair has more than 3 words. Table 5 shows the results.

Training Data	Tune Data (NIST MT03)	Test Data (NIST MT05)
2.08M sentence pairs in News	35.79	34.26
	Tune Data (elec1000-tune)	Test Data (elec1000-test)
	7.93	6.69

Table 3: Experimental results using News training data to test NIST evaluation data and electronic data (numbers denote BLEU score points in percent).

Method	Tune (elec1000-tune)	Test (elec1000-test)
Baseline	7.93	6.69
baseline + in-domain lexicon	10.97	<b>10.87</b>
baseline + in-domain lexicon + in-domain language model	13.72	<b>13.41<sup>++</sup></b>
Transductive Learning	14.13	<b>14.01*</b>

Table 4: Experimental results using News training data, in-domain lexicon, language model and transductive learning. **Bold** figures mean that the results are statistically significant better than the baseline with  $p < 0.01$ , and “<sup>++</sup>” denotes the result is statistically significant better than *baseline+in-domain lexicon*. “\*” means that the result is statistically significant better than 13.41 with  $p < 0.05$ .

Method	Tune (BLEU %)	Test (BLEU %)
BestConfig	13.72	13.41
+phrase pair induction (100k)	14.23	<b>14.06</b>
+phrase pair induction (200k)	14.45	<b>14.24</b>
+phrase pair induction (300k)	14.76	<b>14.83<sup>++</sup></b>
+phrase pair induction (500k)	14.98	<b>15.16<sup>++</sup></b>
+phrase pair induction (1m)	15.11	<b>15.30<sup>++</sup></b>

Table 5: Experimental results of our phrase pair induction method. **Bold** figures denotes the corresponding method significantly outperform the BestConfig with  $p < 0.05$ . **Bold** and *Italic* figures means the results are significantly better than that of BestConfig with  $p < 0.01$ . “<sup>++</sup>” denotes that the corresponding approach performs significantly better than Transductive Learning with  $p < 0.01$ .

Method	Before Filtering	After Filtering
+phrase pair induction (100k)	72,615	8,724
+phrase pair induction (200k)	108,948	12,328
+phrase pair induction (300k)	136,529	17,505
+phrase pair induction (500k)	150,263	19,862
+phrase pair induction (1m)	169,172	21,486

Table 6: the number of phrase pairs induced with different size of monolingual data.

We can see from the table that our method obtains the best translation performance. When using the first 100k sentences for phrase pair induction, it obtains a significant improvement over the BestConfig by 0.65 BLEU points and can outperform the transductive learning method. When we use more monolingual data, the performance becomes even better. The method of phrase pair induction using 300k sentences performs quite well. It outperforms the BestConfig significantly with an improvement of 1.42 BLEU points and it also performs much better than transductive learning method with gains of 0.82 BLEU points. With the monolingual data larger

and larger, the gains become smaller and smaller because the word hit rate gets lower and lower. These experimental results empirically show the effectiveness of our proposed phrase pair induction method.

A question remains that how many new phrase pairs are induced with different size of monolingual data. Here, we give respectively the statistics before and after filtering with the 1000 test sentences. Table 6 shows the statistics. We can see from the table that lots of new phrase pairs can be induced since the source and target monolingual data is in the same domain. However, since the source and target monolingual data is

far from parallel, most of the phrase pairs are not long. For example, in the 108,948 distinct phrase pairs, we find that the phrase pair distribution according to source-side length is (3:50.6%, 4:35.6%, 5:3.3%, 6:9.8%, 7:0.7%). It is easy to see that the phrase pairs whose source-side length longer than 4 account for only a very small part.

## 8 Conclusion and Future Work

This paper proposes a simple but effective method to induce phrase pairs from monolingual data. Given the probabilistic bilingual lexicon and both-side monolingual data in the same domain, the method employs inverted index structure to represent the target-side monolingual data, and induce the translations for each distinct source-side phrase with the help of the bilingual lexicon. We further propose an approach to refine the result phrase pairs to make them more accurate. We also introduce how to estimate the translation and reordering parameters for the induced phrase pairs with monolingual data. Extensive experiments on domain adaptation have shown that our method can significantly outperform previous methods which also focus on exploring the in-domain lexicon and monolingual data.

However, through the analysis we find that our induced phrase pairs still contain some noise, such as the words in source- and target-side of the phrase pair are all aligned but the target-side phrase expresses the different meaning. Furthermore, our proposed method cannot learn expressions which are not lexical translations but are semantic ones. In the future, we will study further on these phenomena and propose new methods to handle these problems.

## Acknowledgments

The research work has been funded by the Hi-Tech Research and Development Program (“863” Program) of China under Grant No. 2011AA01A207, 2012AA011101 and 2012AA011102, and also supported by the Key Project of Knowledge Innovation of Program of Chinese Academy of Sciences under Grant No. KGZD-EW-501. We would also like to thank the anonymous reviewers for their valuable suggestions.

## References

Nicola Bertoldi and Marcello Federico, 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proc. of the Fourth*

*Workshop on Statistical Machine Translation*, pages 182-189.

Mauro Cettolo, Marcello Federico and Nicola Bertoldi, 2010. Mining parallel fragments from comparable texts. In *Proc. of the seventh International Workshop on Spoken Language Translation (IWSLT)*, pages 227-234.

David Chiang, 2007. Hierarchical phrase-based translation. *computational linguistics*, 33 (2). pages 201-228.

David Chiang, 2010. Learning to translate with source and target syntax. In *Proc. of ACL 2010*, pages 1443-1452.

Hal Daumé III and Jagadeesh Jagarlamudi, 2011. Domain adaptation for machine translation by mining unseen words. In *Proc. of ACL-HLT 2011*.

Qing Dou and Kevin Knight, 2012. Large Scale Decipherment for Out-of-Domain Machine Translation. In *Proc. of EMNLP-CONLL 2012*.

Ted Dunning, 1993. Accurate methods for the statistics of surprise and coincidence. *computational linguistics*, 19 (1). pages 61-74.

Pascale Fung and Lo Yuen Yee, 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proc. of ACL-COLING 1998.*, pages 414-420.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang and Ignacio Thayer, 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. of COLING-ACL 2006*, pages 961-968.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick and Dan Klein, 2008. Learning bilingual lexicons from monolingual corpora. In *Proc. of ACL-08: HLT*, pages 771-779.

Liang Huang, Kevin Knight and Aravind Joshi, 2006. A syntax-directed translator with extended domain of locality. In *Proc. of AMTA 2006*, pages 1-8.

Alexandre Klementiev, Ann Irvine, Chris Callison-Burch and David Yarowsky, 2012. Toward statistical machine translation without parallel corpora. In *Proc. of EACL 2012.*, pages 130-140.

Philipp Koehn, 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP 2004.*, pages 388-395, Barcelona, Spain, July 25th-26th, 2004.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin and Evan Herbst, 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL on Interactive Poster and Demonstration Sessions 2007.*, pages 177-180, Prague, Czech Republic, June 27th-30th, 2007.

Philipp Koehn and Kevin Knight, 2002. Learning a translation lexicon from monolingual corpora. In

- Proc. of the ACL-02 workshop on Unsupervised lexical acquisition*, pages 9-16.
- Yang Liu, Qun Liu and Shouxun Lin, 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. of COLING-ACL 2006*, pages 609-616.
- I. Dan Melamed, 2000. Models of translational equivalence among words. *computational linguistics*, 26 (2). pages 221-249.
- Rorbert C. Moore, 2004a. Improving IBM word-alignment model 1. In *Proc. of ACL 2004*.
- Rorbert C. Moore, 2004b. On log-likelihood-ratios and the significance of rare events. In *Proc. of EMNLP 2004.*, pages 333-340.
- Dragos Stefan Munteanu and Daniel Marcu, 2006. Extracting parallel sub-sentential fragments from non-parallel corpora. In *Proc. of ACL-COLING 2006*.
- Malte Nuhn, Arne Mauser and Hermann Ney, 2012. Deciphering Foreign Language by Combining Language Models and Context Vectors. In *Proc. of ACL 2012*.
- Franz Josef Och and Hermann Ney., 2003. A systematic comparison of various statistical alignment models. *computational linguistics*, 29 (1). pages 19-51.
- Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu, 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL 2002.*, pages 311-318.
- Chris Quirk, Raghavendra Udupa and Arul Menezes, 2007. Generative models of noisy translations with applications to parallel fragment extraction. In *Proc. of the Machine Translation Summit XI*, pages 377-384.
- Reinhard Rapp, 1995. Identifying word translations in non-parallel texts. In *Proc. of ACL 1995*, pages 320-322.
- Reinhard Rapp, 1999. Automatic identification of word translations from unrelated English and German corpora. In *Proc. of ACL 1999*, pages 519-526.
- Sujith Ravi and Kevin Knight, 2011. Deciphering foreign language. In *Proc. of ACL 2011.*, pages 12-21.
- Holger Schwenk, 2008. Investigations on largescale lightly-supervised training for statistical machine translation. In *Proc. of IWSLT 2008*, pages 182-189.
- Andreas Stolcke, 2002. SRILM-an extensible language modeling toolkit. In *Proc. of 7th International Conference on Spoken Language Processing*, pages 901-904, Denver, Colorado, USA, September 16th-20th, 2002.
- Nicola Ueffing, Gholamreza Haffari and Anoop Sarkar, 2007. Transductive learning for statistical machine translation. In *Proc. of ACL 2007*.
- Hua Wu, Haifeng Wang and Chengqing Zong, 2008. Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proc. of COLING 2008.*, pages 993-1000.
- Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong, 2011. Simple but effective approaches to improving tree-to-tree model. In *Proc. of MT Summit XIII 2011*, pages 261-268.
- Feifei Zhai, Jiajun Zhang, Yu Zhou and Chengqing Zong, 2012. Tree-based translation without using parse trees. In *Proc. of COLING 2012*, pages 3037-3054.
- Jiajun Zhang, Feifei Zhai and Chengqing Zong, 2011. Augmenting string-to-tree translation models with fuzzy use of the source-side syntax. In *Proc. of EMNLP 2011*, pages 204-215.
- Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan and Sheng Li, 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proc. of ACL-08: HLT*, pages 559-567.

# SenseSpotting: Never let your parallel data tie you to an old domain

Marine Carpuat<sup>1</sup>, Hal Daumé III<sup>2</sup>, Katharine Henry<sup>3</sup>,  
Ann Irvine<sup>4</sup>, Jagadeesh Jagarlamudi<sup>5</sup>, Rachel Rudinger<sup>6</sup>

<sup>1</sup> National Research Council Canada, marine.carpuat@nrc.gc.ca

<sup>2</sup> CLIP, University of Maryland, me@hal3.name

<sup>3</sup> CS, University of Chicago, kehenry@uchicago.edu

<sup>4</sup> CLSP, Johns Hopkins University, anni@jhu.edu

<sup>5</sup> IBM T.J. Watson Research Center, jags@us.ibm.com

<sup>6</sup> CLSP, Johns Hopkins University, rachel.rudinger@aya.yale.edu

## Abstract

Words often gain new senses in new domains. Being able to automatically identify, from a corpus of monolingual text, which word *tokens* are being used in a previously unseen sense has applications to machine translation and other tasks sensitive to lexical semantics. We define a task, SENSESPOTTING, in which we build systems to spot tokens that have new senses in new domain text. Instead of difficult and expensive annotation, we build a gold-standard by leveraging cheaply available parallel corpora, targeting our approach to the problem of domain adaptation for machine translation. Our system is able to achieve F-measures of as much as 80%, when applied to word types it has *never seen before*. Our approach is based on a large set of novel features that capture varied aspects of how words change when used in new domains.

## 1 Introduction

As Magnini et al. (2002) observed, the domain of the text that a word occurs in is a useful signal for performing word sense disambiguation (e.g. in a text about finance, *bank* is likely to refer to a financial institution while in a text about geography, it is likely to refer to a *river bank*). However, in the classic WSD task, ambiguous word types and a set of possible senses are known in advance. In this work, we focus on the setting where we observe texts in two different domains and want to identify words in the second text that have a sense that did not appear in the first text, without any lexical knowledge in the new domain.

To illustrate the task, consider the French noun *rapport*. In the parliament domain, this means

	<i>état</i>	<i>rapport</i>	<i>régime</i>
Govt.	geo. state	report	(political) regime
Medical	state (mind) geo. state	report ratio	diet (political) regime
Science	geo. state	ratio report	(political) regime diet
Movies	geo. state	report	(political) regime diet

Table 1: Examples of French words and their most frequent senses (translations) in four domains.

(and is translated as) “report.” However, in moving to a medical or scientific domain, the word *gains a new sense*: “ratio”, which simply does not exist in the parliament domain. In a science domain, the “report” sense exists, but it is dominated about 12:1 by “ratio.” In a medical domain, the “report” sense remains dominant (about 2:1), but the new “ratio” sense appears frequently.

In this paper we define a new task that we call SENSESPOTTING. The goal of this task is to identify words in a new domain monolingual text that appeared in old domain text but which have a new, previously unseen sense<sup>1</sup>. We operate under the framework of phrase sense disambiguation (Carpuat and Wu, 2007), in which we take automatically align *parallel* data in an old domain to generate an initial old-domain sense inventory. This sense inventory provides the set of “known” word senses in the form of phrasal translations. Concrete examples are shown in Table 1. One of our key contributions is the development of a rich set of features based on monolingual text that are indicative of new word senses.

This work is driven by an application need. When machine translation (MT) systems are applied in a new domain, many errors are a result of: (1) previously unseen (OOV) source language words, or (2) source language words that appear with a new sense and which require new transla-

<sup>1</sup>All features, code, data and raw results are at: [github.com/hal3/IntrinsicPSDEvaluation](https://github.com/hal3/IntrinsicPSDEvaluation)

tions<sup>2</sup> (Carpuat et al., 2012). Given monolingual text in a new domain, OOVs are easy to identify, and their translations can be acquired using dictionary extraction techniques (Rapp, 1995; Fung and Yee, 1998; Schafer and Yarowsky, 2002; Schafer, 2006; Haghghi et al., 2008; Mausam et al., 2010; Daumé III and Jagarlamudi, 2011), or active learning (Bloodgood and Callison-Burch, 2010). However, previously seen (even frequent) words which require new translations are harder to spot.

Because our motivation is translation, one significant point of departure between our work and prior related work (§3) is that we focus on *word tokens*. That is, we are not interested *only* in the question of “has this known word (type) gained a new sense?”, but the much more specific question of “is this particular (token) *occurrence* of this known word being used in a new sense?” Note that for both the dictionary mining setting *and* the active learning setting, it is important to consider words in context when acquiring their translations.

## 2 Task Definition

Our task is defined by two data components. Details about their creation are in §5. First, we need an old-domain sense dictionary, extracted from French-English parallel text (in our case, parliamentary proceedings). Next, we need new-domain monolingual French text (we use medical text, scientific text and movie subtitle text). Given these two inputs, our challenge is to find tokens in the new-domain text that are being used in a new sense (w.r.t. the old-domain dictionary).

We assume that we have access to a small amount of new domain parallel “tuning data.” From this data, we can extract a small new domain dictionary (§5). By comparing this new domain dictionary to the old domain dictionary, we can identify which words have gained new senses. In this way, we turn the SENSESPOTTING problem into a supervised binary classification problem: an example is a French word in context (in the new domain monolingual text) and its label is positive when it is being used in a sense that did not exist in the old domain dictionary. In this task, the classifier is always making predictions on words

---

<sup>2</sup>Sense shifts do not always demand new translations; some ambiguities are preserved across languages. E.g., *fenêtre* can refer to a window of a building or on a monitor, but translates as “window” either way. Our experiments use bilingual data with an eye towards improving MT performance: we focus on words that demand new translations.

*outside* this tuning data on word types it *has never seen before!* From an applied perspective, the assumption of a small amount of parallel data in the new domain is reasonable: if we want an MT system for a new domain, we will likely have some data for system tuning and evaluation.

## 3 Related Work

While word senses have been studied extensively in lexical semantics, research has focused on word sense disambiguation, the task of disambiguating words in context given a predefined sense inventory (e.g., Agirre and Edmonds (2006)), and word sense induction, the task of learning sense inventories from text (e.g., Agirre and Soroa (2007)). In contrast, detecting novel senses has not received as much attention, and is typically addressed within word sense induction, rather than as a distinct SENSESPOTTING task. Novel sense detection has been mostly motivated by the study of language change over time. Most approaches model changes in co-occurrence patterns for *word types* when moving between corpora of old and modern language (Sagi et al., 2009; Cook and Stevenson, 2010; Gulordava and Baroni, 2011).

Since these type-based models do not capture polysemy in the new language, there have been a few attempts at detecting new senses at the token-level as in SENSESPOTTING. Lau et al. (2012) leverage a common framework to address sense induction and disambiguation based on topic models (Blei et al., 2003). Sense induction is framed as learning topic distributions for a word type, while disambiguation consists of assigning topics to word tokens. This model can interestingly be used to detect newly coined senses, which might co-exist with old senses in recent language. Baman and Crane (2011) use parallel Latin-English data to learn to disambiguate Latin words into English senses. New English translations are used as evidence that Latin words have shifted sense. In contrast, the SENSESPOTTING task consists of detecting when senses are unknown in parallel data.

Such novel sense induction methods require manually annotated datasets for the purpose of evaluation. This is an expensive process and therefore evaluation is typically conducted on a very small scale. In contrast, our SENSESPOTTING task leverages automatically word-aligned parallel corpora as a source of annotation for supervision during training and evaluation.

The impact of domain on novel senses has also received some attention. Most approaches operate at the *type*-level, thus capturing changes in the most frequent sense of a word when shifting domains (McCarthy et al., 2004; McCarthy et al., 2007; Erk, 2006; Chan and Ng, 2007). Chan and Ng (2007) notably show that detecting changes in predominant sense as modeled by domain sense priors can improve sense disambiguation, even after performing adaptation using active learning.

Finally, SENSESPOTTING has not been addressed directly in MT. There has been much interest in translation mining from parallel or comparable corpora for *unknown* words, where it is easy to identify which words need translations. In contrast, SENSESPOTTING detects when words have new senses and, thus, frequently a new translation. Work on active learning for machine translation has focused on collecting translations for longer unknown segments (e.g., Bloodgood and Callison-Burch (2010)). There has been some interest in detecting which phrases that are hard to translate for a given system (Mohit and Hwa, 2007), but difficulties can arise for many reasons: SENSESPOTTING focuses on a single problem.

## 4 New Sense Indicators

We define features over both *word types* and *word tokens*. In our classification setting, each instance consists of a French word token in context. Our *word type* features ignore this context and rely on statistics computed over our entire new domain corpus. In contrast, our *word token* features consider the context of the particular instance of the word. If it were the case that only one sense existed for all word tokens of a particular type within a single domain, we would expect our word type features to be able to spot new senses without the help of the word token features. However, in fact, even within a single domain, we find that often a word type is used with several senses, suggesting that word token features may also be useful.

### 4.1 Type-level Features

**Lexical Item Frequency Features** A very basic property of the new domain that we hope to capture is that word frequencies change, and such changes might be indicative of a domain shift. As such, we compute unigram log probabilities (via smoothed relative frequencies) of each word under consideration in the old domain and the new domain. We then add as features these two log

probabilities as well as their difference. These are our **Type:RelFreq** features.

**N-gram Probability Features** The goal of the **Type:NgramProb** feature is to capture the fact that “unusual contexts” might imply new senses. To capture this, we can look at the log probability of the word under consideration given its N-gram context, both according to an old-domain language model (call this  $\ell_{ng}^{old}$ ) and a new-domain language model (call this  $\ell_{ng}^{new}$ ). However, we do not simply want to capture unusual words, but words that are unlikely in context, so we also need to look at the respective unigram log probabilities:  $\ell_{ug}^{old}$  and  $\ell_{ug}^{new}$ . From these four values, we compute corpus-level (and therefore type-based) statistics of the new domain n-gram log probability ( $\ell_{ng}^{new}$ , the difference between the n-gram probabilities in each domain ( $\ell_{ng}^{new} - \ell_{ng}^{old}$ ), the difference between the n-gram and unigram probabilities in the new domain ( $\ell_{ng}^{new} - \ell_{ug}^{new}$ ), and finally the combined difference:  $\ell_{ng}^{new} - \ell_{ug}^{new} + \ell_{ug}^{old} - \ell_{ng}^{old}$ ). For each of these four values, we compute the following type-based statistics over the monolingual text: mean, standard deviation, minimum value, maximum value and sum. We use trigram models.

**Topic Model Feature** The intuition behind the topic model feature is that if a word’s distribution over topics changes when moving into a new domain, it is likely to also gain a new sense. For example, suppose that in our old domain, the French word *enceinte* is only used with the sense “wall,” but in our new domain, *enceinte* may have senses corresponding to either “wall” or to “pregnant.” We would expect to see this reflected in *enceinte*’s distribution over topics: the topic that places relatively high probabilities on words such as “bébé” (English “baby”) and *enfant* (English “child”) will also place a high probability on *enceinte* when trained on new domain data. In the old domain, however, we would not expect a similar topic (if it exists) to give a high probability to *enceinte*. Based on this intuition, for all words  $w$ , where  $T_o$  and  $T_n$  are the set of old and new topics and  $P_o$  and  $P_n$  are the old and new distributions defined over them, respectively, and  $cos$  is the cosine similarity between a pair of topics, we define the feature **Type:TopicSim**:  $\sum_{t \in T_n, t' \in T_o} P_n(t|w)P_o(t'|w)cos(t, t')$ . For a word  $w$ , the feature value will be high if, for each new domain topic  $t$  that places high probability on  $w$ , there is an old domain topic  $t'$  that

is similar to  $t$  and also places a high probability on  $w$ . Conversely, if no such topic exists, the score will be low, indicating the word has gained a new sense. We use the online LDA (Blei et al., 2003; Hoffman et al., 2010), implemented in <http://hunch.net/~vw/> to compute topics on the two domains separately. We use 100 topics.

**Context Feature** It is expected that words acquiring new senses will tend to neighbor different sets of words (e.g. different arguments, prepositions, parts of speech, etc.). Thus, we define an additional type level feature to be the ratio of the number of new domain n-grams (up to length three) that contain word  $w$  and which do not appear in the old domain to the total number of new domain n-grams containing  $w$ . With  $N_w$  indicating the set of n-grams in the new domain which contain  $w$ ,  $O_w$  indicating the set of n-grams in the old domain which contain  $w$ , and  $|N_w - O_w|$  indicating the n-grams which contain  $w$  and appear in the new but not the old domain, we define **Type:Context** as  $\frac{|N_w - O_w|}{|N_w|}$ . We do not count n-grams containing OOVs, as they may simply be instances of applying the same sense of a word to a new argument

## 4.2 Token-level Features

**N-gram Probability Features** Akin to the N-gram probability features at the type level (namely, **Token:NgramProb**), we compute the same values at the token level (new/old domain and unigram/trigram). Instead of computing statistics over the entire monolingual corpus, we use the instantaneous values of these features for the token under consideration. The six features we construct are: unigram (and trigram) log probabilities in the old domain, the new domain, and their difference.

**Context Features** Following the type-level n-gram feature, we define features for a particular word *token* based on its n-gram context. For token  $w_i$ , in position  $i$  in a given sentence, we consider its context words in a five word window:  $w_{i-2}$ ,  $w_{i-1}$ ,  $w_{i+1}$ , and  $w_{i+2}$ . For each of the four contextual words in positions  $p = \{-2, -1, 1, 2\}$ , relative to  $i$ , we define the following feature, **Token:CtxCnt**:  $\log(c_{w_p})$  where  $c_{w_p}$  is the number of times word  $w_p$  appeared in position  $p$  relative to  $w_i$  in the OLD-domain data. We also define a single feature which is the percent of the four contextual words which had been seen in the OLD-domain data, **Token:Ctx%**.

**Token-Level PSD Features** These features aim to capture generalized characteristics of a context.

Towards this end, first, we pose the problem as a phrase sense disambiguation (PSD) problem over the known sense inventory. Given a source word in a context, we train a classifier to predict the most likely target translation. The ground truth labels (target translation for a given source word) for this classifier are generated from the phrase table of the old domain data. We use the same set of features as in Carpuat and Wu (2007). Second, given a source word  $s$ , we use this classifier to compute the probability distribution of target translations ( $p(t|s)$ ). Subsequently, we use this probability distribution to define new features for the SENSESPOTTING task. The idea is that, if a word is used in one of the known senses then its context must have been seen previously and hence we hope that the PSD classifier outputs a spiky distribution. On the other hand, if the word takes a new sense then hopefully it is used in an unseen context resulting in the PSD classifier outputting an uniform distribution. Based on this intuition, we add the following features: **MaxProb** is the maximum probability of any target translation:  $\max_t p(t|s)$ . **Entropy** is the entropy of the probability distribution:  $-\sum_t p(t|s) \log p(t|s)$ . **Spread** is the difference between maximum and minimum probabilities of the probability distribution:  $(\max_t p(t|s) - \min_t p(t|s))$ . **Confusion** is the uncertainty in the most likely prediction given the source token:  $\frac{\text{median}_t p(t|s)}{\max_t p(t|s)}$ . The use of median in the numerator rather than the second best is motivated by the observation that, in most cases, top ranked translations are of the same sense but differ in morphology.

We train the PSD classifier in two modes: 1) a single global classifier that predicts the target translation given any source word; 2) a local classifier for each source word. When training the global PSD classifier, we include some lexical features that depend on the source word. For both modes, we use real valued and binned features giving rise to four families of features **Token:G-PSD**, **Token:G-PSDBin**, **Token:L-PSD** and **Token:L-PSDBin**.

**Prior vs. Posterior PSD Features** When the PSD classifier is trained in the second mode, i.e. one classifier per word type, we can define additional features based on the prior (with out the word context) and posterior (given the word's context) probability distributions output by the classifier, i.e.  $p_{\text{prior}}(t|s)$  and  $p_{\text{post}}(t|s)$  respec-

Domain	Sentences	Lang	Tokens	Types
Hansard	8,107,356	fr	161,695,309	191,501
		en	144,490,268	186,827
EMEA	472,231	fr	6,544,093	34,624
		en	5,904,296	29,663
Science	139,215	fr	4,292,620	117,669
		en	3,602,799	114,217
Subs	19,239,980	fr	154,952,432	361,584
		en	174,430,406	293,249

Table 2: Basic characteristics of the parallel data.

tively. We compute the following set of features referred to as **Token:PSDRatio**: **SameMax** checks if both the prior and posterior distributions have the same translation as the most likely translation. **SameMin** is same as the above feature but check if the least likely translation is same. **X-OR\_MinMax** is the exclusive-OR of **SameMax** and **SameMin** features. **KL** is the KL-divergence between the two distributions. Since KL-divergence is asymmetric, we use  $KL(p_{\text{prior}}||p_{\text{post.}})$  and  $KL(p_{\text{post.}}||p_{\text{prior}})$ . **MaxNorm** is the ratio of maximum probabilities in prior and posterior distributions. **SpreadNorm** is the ratio of spread of the prior and posterior distributions, where spread is the difference between maximum and minimum probabilities of the distribution as defined earlier. **ConfusionNorm** is the ratio of confusion of the prior and posterior distributions, where confusion is defined as earlier.

## 5 Data and Gold Standard

The first component of our task is a parallel corpus of old domain data, for which we use the French-English Hansard parliamentary proceedings (<http://www.parl.gc.ca>). From this, we extract an old domain sense dictionary, using the Moses MT framework (Koehn et al., 2007). This defines our old domain sense dictionary. For new domains, we use three sources: (1) the EMEA medical corpus (Tiedemann, 2009), (2) a corpus of scientific abstracts, and (3) a corpus of translated movie subtitles (Tiedemann, 2009). Basic statistics are shown in Table 2. In all parallel corpora, we normalize the English for American spelling.

To create the gold standard *truth*, we followed a lexical sample approach and collected a set of 300 “representative types” that are interesting to evaluate on, because they have multiple senses within a single domain or whose senses are likely to change in a new domain. We used a semi-automatic approach to identify representative types. We first used the phrase table from

	Parallel		Repr. Types	Repr. Tokens	% New Sense
	Sents	fr-tok			
EMEA	24k	270k	399	35,266	52.0%
Science	22k	681k	425	8,355	24.3%
Subs	36k	247k	388	22,598	43.4%

Table 3: Statistics about representative words and the size of the development sets. The columns show: the total amount of parallel development data (# of sentences and tokens in French), # of representative types that appear in this corpus, the corresponding # of tokens, and the percentage of these tokens that correspond to “new senses.”

the Moses output to rank phrases in each domain using TF-IDF scores with Okapi BM25 weighting. For each of the three new domains (EMEA, Science, and Subs), we found the intersection of phrases between the old and the new domain. We then looked at the different translations that each had in the phrase table and a French speaker selected a subset that have multiple senses.<sup>3</sup>

In practice, we limited our set almost entirely to source *words*, and included only a single multi-word phrase, *vue des enfants*, which usually translates as “for children” in the old domain but almost always translates as “sight of children” in the EMEA domain (as in “. . . should be kept out of *the sight of children*”). Nothing in the way we have defined, approached, or evaluated the SENSESPOTTING task is dependent on the use of representative words instead of longer representative phrases. We chose to consider mostly source language words for simplicity and because it was easier to identify good candidate words.

In addition to the manually chosen words, we also identified words where the translation with the highest lexical weight varied in different domains, with the intuition being that are the words that are likely to have acquired a new sense. The top 200 words from this were added to the manually selected representative words to form a list of 450. Table 3 shows some statistics about these words across our three test domains.

## 6 Experiments

### 6.1 Experimental setup

Our goal in evaluation is to be able to understand what our approach is realistically capable of. One challenge is that the distribution

<sup>3</sup>In order to create the *evaluation data*, we used both sides of the full parallel text; we do *not* use the English side of the parallel data for actually building systems.

of representative words is highly skewed.<sup>4</sup> We present results in terms of area under the ROC curve (AUC),<sup>5</sup> micro-averaged precision/recall/f-measure and macro-averaged precision/recall/f-measure. For macro-averaging, we compute a single confusion matrix over all the test data and determining P/R/F from that matrix. For micro-averaging, we compute a separate confusion matrix *for each word type* on the French side, compute P/R/F for each of these separately, and then average the results. (Thus, micro-F is not a function of micro-P and micro-R.) The AUC and macro-averaged scores give a sense of how well the system is doing on a type-level basis (essentially weighted by type frequency), while the micro-averaged scores give a sense as to how well the system is doing on individual types, not taking into account their frequencies.

For most of our results, we present standard deviations to help assess significance ( $\pm 2\sigma$  is roughly a 90% confidence interval). For our results, in which we use new-domain training data, we compute these results via 16-fold cross validation. The folds are split *across types so the system is never being tested on a word type that it has seen before*. We do this because it more closely resembles our application goals. We do 16-fold for convenience, because we divide the data into binary folds recursively (thus having a power-of-two is easier), with an attempt to roughly balance the size of the training sets in each fold (this is tricky because of the skewed nature of the data). This entire 16-fold cross-validation procedure is repeated 10 times and averages and standard deviations are over the 160 replicates.

We evaluate performance using our type-level features only, TYPEONLY, our token-level features only, TOKENONLY, and using both our type and our token level features, ALLFEATURES.

We compare our results with two baselines: RANDOM and CONSTANT. RANDOM predicts new-sense or not-new-sense randomly and with equal probability. CONSTANT always predicts new-sense, achieving 100% recall and a macro-level precision that is equal to the percent of representative words which do have a new sense, modulo cross-validation splits (see Table 3). Addi-

<sup>4</sup>The most frequent (*voie*) appears 3881 times; there are 60 singleton words on average across the three new domains.

<sup>5</sup>AUC is the probability that the classifier will assign a higher score to a randomly chosen positive example than to a randomly chosen negative example (Wikipedia, 2013).

tionally, we compare our results with a type-level oracle, TYPEORACLE. For all tokens of a given word type, the oracle predicts the majority label (new-sense or not-new-sense) for that word type. These results correspond to an upper bound for the TYPEONLY experiments.

## 6.2 Classification Setup

For all experiments, we use a linear classifier trained by stochastic gradient descent to optimize logistic loss. We also did some initial experiments on development data using boosted decision trees instead and other loss functions (hinge loss, squared loss), but they never performed as well. In all cases, we perform 20 passes over the training data, using development data to perform early stopping (considered at the end of each pass). We also use development data to tune a regularizer (either  $\ell_1$  or  $\ell_2$ ) and its regularization weight.<sup>6</sup> Finally, all real valued features are automatically bucketed into 10 consecutive buckets, each with (approximately) the same number of elements. Each learner uses a small amount of development data to tune a threshold on scores for predicting new-sense or not-a-new-sense, using macro F-measure as an objective.

## 6.3 Result Summary

Table 4 shows our results on the SENSESPOTTING task. Classifiers based on the features that we defined outperform both baselines in all macro-level evaluations for the SENSESPOTTING task. Using AUC as an evaluation metric, the TOKENONLY, TYPEONLY, and ALLFEATURES models performed best on EMEA, Science, and Subtitles data, respectively. Our token-level features perform particularly poorly on the Science and Subtitles data. Although the model trained on only those features achieves reasonable precision (72.59 and 70.00 on Science and Subs, respectively), its recall is very low (20.41 and 35.15), indicating that the model classifies many new-sense words as not-new-sense. Most of our token-level features capture the intuition that when a word token appears in new or infrequent contexts, it is likely to have gained a new sense. Our results indicate that this intuition was more fruitful for EMEA than for Science or Subs.

In contrast, the type-only features (TYPEONLY)

<sup>6</sup>We use <http://hunch.net/~vw/> version 7.1.2, and run it with the following arguments that affect learning behavior: `--exact_adaptive_norm --power_t 0.5`

	AUC	Macro			Micro		
		P	R	F	P	R	F
<b>EMEA</b>							
RANDOM	50.34 ± 0.60	51.24 ± 0.59	50.09 ± 1.18	50.19 ± 0.75	47.04 ± 0.60	56.07 ± 1.99	37.27 ± 0.91
CONSTANT	50.00 ± 0.00	50.99 ± 0.00	<b>100.0</b> ± 0.00	67.09 ± 0.00	45.80 ± 0.00	<b>100.0</b> ± 0.00	<b>52.30</b> ± 0.00
TYPEONLY	55.91 ± 1.13	69.76 ± 3.45	43.13 ± 1.42	41.61 ± 1.07	<b>77.92</b> ± 2.04	50.12 ± 2.35	31.26 ± 0.63
TYPEORACLE	88.73 ± 0.00	87.32 ± 0.00	86.76 ± 0.00	87.04 ± 0.00	90.01 ± 0.00	67.46 ± 0.00	59.39 ± 0.00
TOKENONLY	78.80 ± 0.52	<b>69.83</b> ± 1.59	75.58 ± 2.61	69.40 ± 1.92	59.03 ± 1.70	62.53 ± 1.66	43.39 ± 0.94
ALLFEATURES	<b>79.60</b> ± 1.20	68.11 ± 1.19	<b>79.84</b> ± 2.27	<b>71.64</b> ± 1.83	55.28 ± 1.11	<b>71.50</b> ± 1.62	<b>46.83</b> ± 0.62
<b>Science</b>							
RANDOM	50.18 ± 0.78	24.48 ± 0.57	50.32 ± 1.33	32.92 ± 0.79	46.99 ± 0.51	60.32 ± 1.06	34.72 ± 1.03
CONSTANT	50.00 ± 0.00	24.34 ± 0.00	<b>100.0</b> ± 0.00	39.15 ± 0.00	44.39 ± 0.00	<b>100.0</b> ± 0.00	<b>50.44</b> ± 0.00
TYPEONLY	<b>77.06</b> ± 1.23	<b>66.07</b> ± 2.80	36.28 ± 4.10	34.50 ± 4.06	<b>84.97</b> ± 0.82	36.81 ± 2.33	24.22 ± 1.70
TYPEORACLE	88.76 ± 0.00	78.43 ± 0.00	69.29 ± 0.00	73.54 ± 0.00	84.19 ± 0.00	67.41 ± 0.00	52.67 ± 0.00
TOKENONLY	66.62 ± 0.47	60.50 ± 3.11	28.05 ± 2.06	30.81 ± 2.75	76.21 ± 1.78	36.57 ± 2.23	24.68 ± 1.36
ALLFEATURES	73.91 ± 0.66	50.59 ± 2.08	<b>60.60</b> ± 2.04	<b>47.54</b> ± 1.52	66.72 ± 1.19	<b>62.30</b> ± 1.36	<b>40.22</b> ± 1.03
<b>Subs</b>							
RANDOM	50.26 ± 0.69	42.47 ± 0.60	50.17 ± 0.84	45.68 ± 0.68	52.18 ± 1.32	54.63 ± 2.01	39.87 ± 2.10
CONSTANT	50.00 ± 0.00	42.51 ± 0.00	<b>100.0</b> ± 0.00	<b>59.37</b> ± 0.00	50.63 ± 0.00	<b>100.0</b> ± 0.00	<b>58.67</b> ± 0.00
TYPEONLY	67.16 ± 0.73	<b>76.41</b> ± 1.51	31.91 ± 3.15	36.37 ± 2.58	<b>90.03</b> ± 0.61	34.78 ± 1.12	26.20 ± 0.61
TYPEORACLE	81.35 ± 0.00	83.12 ± 0.00	70.23 ± 0.00	76.12 ± 0.00	90.62 ± 0.00	52.37 ± 0.00	44.43 ± 0.00
TOKENONLY	63.30 ± 0.99	63.17 ± 2.31	45.38 ± 2.07	43.30 ± 1.29	76.38 ± 1.68	49.70 ± 1.76	37.92 ± 1.20
ALLFEATURES	<b>69.26</b> ± 0.60	63.48 ± 1.77	<b>56.22</b> ± 2.66	<b>52.78</b> ± 1.96	67.55 ± 0.83	<b>62.18</b> ± 1.45	<b>43.85</b> ± 0.90

Table 4: Complete SENSESPOTTING results for all domains. The scores are from cross-validation on a single domain; in all cases, higher is better. Two standard deviations of performance over the cross-validation are shown in small type. For all domains and metrics, the highest (not necessarily statistically significant) non-oracle results are bolded.

are relatively weak for predicting new senses on EMEA data but stronger on Subs (TYPEONLY AUC performance is higher than both baselines) and even stronger on Science data (TYPEONLY AUC and f-measure performance is higher than both baselines as well as the ALLFEATURES model). In our experience with the three datasets, we know that the Science data, which contains abstracts from a wide variety of scientific disciplines, is the most diverse, followed by the Subs data, and then EMEA, which mostly consists of text from drug labels and tends to be quite repetitive. Thus, it makes sense that type-level features would be the most informative for the least homogeneous dataset. Representative words in scientific text are likely to appear in variety of contexts, while in the EMEA data they may only appear in a few, making it easier to contrast them with the distributions observed in the old domain data.

For all domains, in micro-level evaluation, our models fail to outperform the CONSTANT baseline. Recall that the micro-level evaluation computes precision, recall, and f-measure for all word tokens of a given word type and then averages across word types. We observe that words that are less frequent in both the old and the new domains are more likely to have a new sense than more frequent words, which causes the CONSTANT base-

line to perform reasonably well. In contrast, it is more difficult for our models to make good predictions for less frequent words. A low frequency in the new domain makes type level features (estimated over only a few instances) noisy and unreliable. Similarly, a low frequency in the old domain makes the our token level features, which all contrast with old domain instances of the word type.

#### 6.4 Feature Ablation

In the previous section, we observed that (with one exception) both Type-level and Token-level features are useful in our task (in some cases, essential). In this section, we look at finer-grained feature distinctions through a process of feature ablation. In this setting, we begin with *all features* in a model and *remove* one feature at a time, always removing the feature that hurts performance least. For these experiments, we determine which feature to remove using AUC. Note that we’re actually able to beat (by 2-4 points AUC) the scores from Table 4 by removing features!

The results here are somewhat mixed. In EMEA and Science, one can actually get by (according to AUC) with very few features: just two (Type:NgramProband Type:Context) are sufficient to achieve optimal AUC scores. To get higher Macro-F scores requires nearly all the features, though this is partially due to the choice of

EMEA	AUC	MacF	Science	AUC	MacF	Subs	AUC	MacF
ALLFEATURES	79.60	71.64	ALLFEATURES	73.91	47.54	ALLFEATURES	69.26	52.78
-Token:L-PSDBin	77.09	70.50	-Token:L-PSDBin	76.26	53.69	-Type:NgramProb	69.13	53.33
-Type:RelFreq	78.43	72.19	-Token:G-PSD	77.04	53.56	-Token:G-PSDBin	70.23	54.72
-Token:G-PSD	<b>79.66</b>	72.11	-Token:G-PSDBin	77.44	54.54	-Token:CtxCnt	71.23	58.35
-Type:Context	<b>79.66</b>	72.45	-Token:L-PSD	77.85	56.05	-Token:L-PSDBin	72.07	57.85
-Token:Ctx%	78.91	<b>73.37</b>	-Token:PSDRatio	77.92	<b>57.34</b>	-Token:G-PSD	72.17	57.33
-Type:TopicSim	78.05	71.33	-Token:CtxCnt	77.85	54.42	-Type:TopicSim	<b>72.31</b>	58.41
-Token:CtxCnt	76.90	71.72	-Type:Context	<b>78.17</b>	55.45	-Token:Ctx%	72.17	56.17
-Token:L-PSD	76.03	73.35	-Token:Ctx%	78.06	55.04	-Token:NgramProb	71.35	<b>59.26</b>
-Type:NgramProb	73.32	69.54	-Type:TopicSim	77.83	54.57	-Token:PSDRatio	70.33	46.88
-Token:G-PSDBin	74.41	69.76	-Token:NgramProb	76.98	51.02	-Token:L-PSD	69.05	53.31
-Token:NgramProb	69.78	68.89	-Type:RelFreq	74.25	49.57	-Type:RelFreq	65.25	48.22
-Token:PSDRatio	48.38	3.45	-Type:NgramProb	50.00	0.00	-Type:Context	50.00	0.00

Table 5: Feature ablation results for all three corpora. Selection criteria is AUC, but Macro-F is presented for completeness. Feature selection is run independently on each of the three datasets. The features toward the *bottom* were the first selected.

	AUC	Macro-F	Micro-F
<b>EMEA</b>			
TYPEONLY	71.43 ± 0.94	52.62 ± 3.41	38.67 ± 1.35
TOKENONLY	<b>73.75</b> ± 1.11	<b>67.77</b> ± 4.18	45.49 ± 3.96
ALLFEATURES	72.19 ± 4.07	67.26 ± 7.88	<b>49.29</b> ± 3.55
XV-ALLFEATURES	79.60 ± 1.20	71.64 ± 1.83	46.83 ± 0.62
<b>Science</b>			
TYPEONLY	<b>75.19</b> ± 0.89	<b>51.53</b> ± 2.55	37.14 ± 4.41
TOKENONLY	71.24 ± 1.45	47.27 ± 1.11	40.48 ± 1.84
ALLFEATURES	74.14 ± 0.93	48.86 ± 3.94	<b>43.20</b> ± 3.16
XV-ALLFEATURES	73.91 ± 0.66	47.54 ± 1.52	40.22 ± 1.03
<b>Subs</b>			
TYPEONLY	60.90 ± 1.47	39.21 ± 14.78	24.77 ± 2.78
TOKENONLY	<b>62.00</b> ± 1.16	49.74 ± 6.30	<b>42.95</b> ± 3.92
ALLFEATURES	60.12 ± 2.11	<b>50.16</b> ± 8.63	38.56 ± 5.20
XV-ALLFEATURES	69.26 ± 0.60	52.78 ± 1.96	43.85 ± 0.90

Table 6: Cross-domain test results on the SENSESPOTTING task. Two standard deviations are shown in small type. Only AUC, Macro-F and Micro-F are shown for brevity.

AUC as the measure on which to ablate. It’s quite clear that for Science, all the useful information is in the type-level features, a result that echoes what we saw in the previous section. While for EMEA and Subs, both type- and token-level features play a significant role. Considering the six most useful features in each domain, the ones that pop out as frequently most useful are the global PSD features, the ngram probability features (either type- or token-based), the relative frequency features and the context features.

## 6.5 Cross-Domain Training

One disadvantage to the previous method for evaluating the SENSESPOTTING task is that it requires parallel data in a new domain. Suppose we have *no* parallel data in the new domain at all, yet still want to attack the SENSESPOTTING task. One option is

to train a system on domains for which we *do* have parallel data, and then apply it in a new domain. This is precisely the setting we explore in this section. Now, instead of performing cross-validation in a single domain (for instance, Science), we take the union of *all* of the training data in the other domains (e.g., EMEA and Subs), train a classifier, and then apply it to Science. This classifier will almost certainly be worse than one trained on NEW (Science) but does not require *any* parallel data in that domain. (Hyperparameters are chosen by development data from the OLD union.)

The results of this experiment are shown in Table 6. We include results for TOKENONLY, TYPEONLY and ALLFEATURES; all of these are trained in the cross-domain setting. To ease comparison to the results that do not suffer from domain shift, we also present “XV-ALLFEATURES”, which are results copied from Table 4 in which parallel data from NEW is used. Overall, there is a drop of about 7.3% absolute in AUC, moving from XV-ALLFEATURES to ALLFEATURES, including a small improvement in Science (likely because Science is markedly smaller than Subs, and “more difficult” than EMEA with many word types).

## 6.6 Detecting Most Frequent Sense Changes

We define a second, related task: MOSTFREQUENSENSECHANGE. In this task, instead of predicting if a given word token has a sense which is brand new with respect to the old domain, we predict whether it is being used with a sense which is not the one that was observed *most frequently* in the old domain. In our EMEA, Science, and Subtitles data, 68.2%, 48.3%, and 69.6% of word tokens’ predominant sense changes.

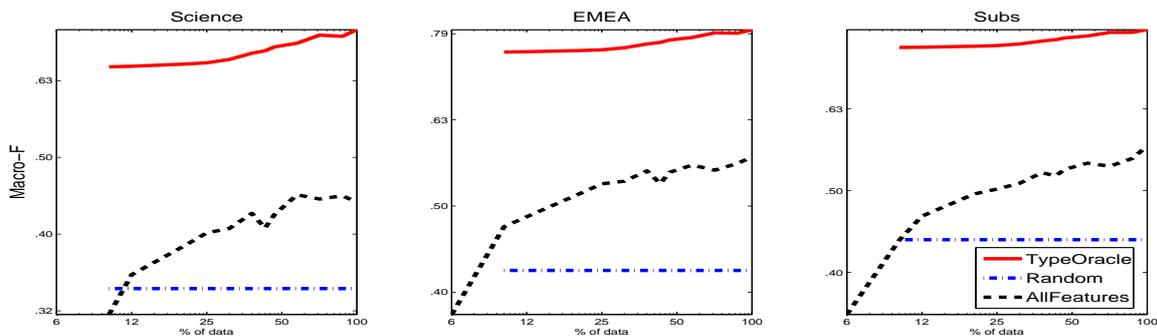


Figure 1: Learning curves for the three domains. X-axis is percent of data used, Y-axis is Macro-F score. Both axes are in log scale to show the fast rate of growth. A horizontal bar corresponding to random predictions, and the TYPEORACLE results are shown for comparison.

	AUC	Macro-F	Micro-F
<b>EMEA</b>			
RANDOM	50.54 ± 0.41	58.23 ± 0.34	49.69 ± 0.85
CONSTANT	50.00 ± 0.00	<b>82.15</b> ± 0.00	<b>74.43</b> ± 0.00
TYPEONLY	55.05 ± 1.00	67.45 ± 1.35	65.72 ± 0.59
TYPEORACLE	88.36 ± 0.00	90.64 ± 0.00	77.46 ± 0.00
TOKENONLY	<b>66.42</b> ± 1.07	80.27 ± 0.50	68.96 ± 0.58
ALLFEATURES	58.64 ± 3.45	<b>80.57</b> ± 0.45	<b>69.40</b> ± 0.51
<b>Science</b>			
RANDOM	50.13 ± 0.78	49.05 ± 0.82	48.19 ± 1.47
CONSTANT	50.00 ± 0.00	<b>65.21</b> ± 0.00	<b>73.22</b> ± 0.00
TYPEONLY	68.32 ± 1.05	54.70 ± 2.35	57.04 ± 1.52
TYPEORACLE	91.41 ± 0.00	86.71 ± 0.00	74.26 ± 0.00
TOKENONLY	<b>68.49</b> ± 0.59	62.76 ± 0.89	64.40 ± 1.08
ALLFEATURES	68.31 ± 0.93	<b>64.73</b> ± 1.93	<b>67.20</b> ± 1.65
<b>Subs</b>			
RANDOM	50.27 ± 0.27	56.93 ± 0.29	50.93 ± 1.11
CONSTANT	50.00 ± 0.00	<b>79.96</b> ± 0.00	<b>76.26</b> ± 0.00
TYPEONLY	<b>60.36</b> ± 0.90	67.78 ± 1.98	61.58 ± 1.78
TYPEORACLE	82.16 ± 0.00	87.96 ± 0.00	73.87 ± 0.00
TOKENONLY	59.49 ± 1.04	<b>77.79</b> ± 0.82	<b>73.51</b> ± 0.68
ALLFEATURES	54.97 ± 0.89	77.30 ± 1.58	72.29 ± 1.68

Table 7: Cross-validation results on the MOSTFREQSENSECHANGE task. Two standard deviations are shown in small type.

We use the same set of features and learning framework to generate and evaluate models for this task. While the SENSESPOTTING task has MT utility in suggesting which new domain words demand a new translation, the MOSTFREQSENSECHANGE task has utility in suggesting which words demand a new translation probability distribution when shifting to a new domain. Table 7 shows the results of our MOSTFREQSENSECHANGE task experiments.

Results on the MOSTFREQSENSECHANGE task are somewhat similar to those for the SENSESPOTTING task. Again, our models perform better under a macro-level evaluation than under a micro-level evaluation. However, in contrast to the SENSESPOTTING results, token-level features

perform quite well on their own for all domains. It makes sense that our token level features have a better chance of success on this task. The important comparison now is between a new domain token in context and the *majority* of the old domain tokens of the same word type. This comparison is likely to be more informative than when we are equally interested in identifying overlap between the current token and any old domain senses. Like the SENSESPOTTING results, when doing a micro-level evaluation, our models do not perform as well as the CONSTANT baseline, and, as before, we attribute this to data sparsity.

## 6.7 Learning Curves

All of the results presented so far use classifiers trained on instances of representative types (i.e. “representative tokens”) extracted from fairly large new domain parallel corpora (see Table 3), consisting of between 22 and 36 thousand parallel sentences, which yield between 8 and 35 thousand representative tokens. Although we expect some new domain parallel tuning data to be available in most MT settings, we would like to know how many representative types are required to achieve good performance on the SENSESPOTTING task. Figure 6.5 shows learning curves over the number of representative tokens that are used to train SENSESPOTTING classifiers. In fact, only about 25-50% of the data we used is really necessary to achieve the performance observed before.

**Acknowledgments** We gratefully acknowledge the support of the JHU summer workshop program (and its funders), the entire DAMT team (<http://hal3.name/DAMT/>), Sanjeev Khudanpur, support from the NRC for Marine Carpuat, as well as DARPA CSSG Grant D11AP00279 for Hal Daumé III and Jagadeesh Jagarlamudi.

## References

- E. Agirre and P.G. Edmonds. 2006. *Word Sense Disambiguation: Algorithms and Applications*. Text, Speech, and Language Technology Series. Springer Science+Business Media B.V.
- Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 7–12.
- David Bamman and Gregory Crane. 2011. Measuring historical word sense variation. In *Proceedings of the 2011 Joint International Conference on Digital Libraries (JCDL 2011)*, pages 1–10.
- D. Blei, A. Ng, and M. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 3.
- Michael Bloodgood and Chris Callison-Burch. 2010. Bucking the trend: Large-scale cost-focused active learning for statistical machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 854–864, Uppsala, Sweden, July. Association for Computational Linguistics.
- Marine Carpuat and Dekai Wu. 2007. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 61–72, Prague, June.
- Marine Carpuat, Hal Daumé III, Alexander Fraser, Chris Quirk, Fabienne Braune, Ann Clifton, Ann Irvine, Jagadeesh Jagarlamudi, John Morgan, Majid Razmara, Aleš Tamchyna, Katharine Henry, and Rachel Rudinger. 2012. Domain adaptation in machine translation: Final report. In *2012 Johns Hopkins Summer Workshop Final Report*.
- Yee Seng Chan and Hwee Tou Ng. 2007. Domain adaptation with active learning for word sense disambiguation. In *Proceedings of the Association for Computational Linguistics*.
- Paul Cook and Suzanne Stevenson. 2010. Automatically identifying changes in the semantic orientation of words. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, pages 28–34, Valletta, Malta.
- Hal Daumé III and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.
- Katrin Erk. 2006. Unknown word sense detection as outlier detection. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 128–135.
- Pascale Fung and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.
- Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the google books ngram corpus. In *Proceedings of the GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics*, pages 67–71, Edinburgh, UK, July. Association for Computational Linguistics.
- Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.
- Matthew Hoffman, David Blei, and Francis Bach. 2010. Online learning for latent dirichlet allocation. In *Advances in Neural Information Processing Systems (NIPS)*.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.
- Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, Timothy Baldwin, and Lexical Computing. 2012. Word sense induction for novel sense detection. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 591–601. Citeseer.
- Bernardo Magnini, Carlo Strapparava, Giovanni Pezzulo, and Alfio Gliozzo. 2002. The role of domain information in word sense disambiguation. *Natural Language Engineering*, 8(04):359–373.
- Mausam, Stephen Soderland, Oren Etzioni, Daniel S. Weld, Kobi Reiter, Michael Skinner, Marcus Sammer, and Jeff Bilmes. 2010. Panlingual lexical translation via probabilistic inference. *Artificial Intelligence*, 174:619–637, June.
- Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 279. Association for Computational Linguistics.
- Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2007. Unsupervised acquisition of predominant word senses. *Computational Linguistics*, 33(4):553–590.

- Behrang Mohit and Rebecca Hwa. 2007. Localization of difficult-to-translate phrases. In *proceedings of the 2nd ACL Workshop on Statistical Machine Translations*.
- Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*.
- Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic density analysis: Comparing word meaning across time and phonetic space. In *Proceedings of the EACL 2009 Workshop on GEMS: GEometrical Models of Natural Language Semantics*, pages 104–111, Athens, Greece, March.
- Charles Schafer and David Yarowsky. 2002. Inducing translation lexicons via diverse similarity measures and bridge languages. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*.
- Charles Schafer. 2006. *Translation Discovery Using Diverse Similarity Measures*. Ph.D. thesis, Johns Hopkins University.
- Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing (RANLP)*.
- Wikipedia. 2013. Receiver operating characteristic. [http://en.wikipedia.org/wiki/Receiver\\_operating\\_characteristic#Area\\_Under\\_the\\_Curve](http://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_Under_the_Curve), February.

# BRAINSUP: Brainstorming Support for Creative Sentence Generation

**Gözde Özbal**

FBK-irst  
Trento, Italy  
gozbalde@gmail.com

**Daniele Pighin**

Google Inc.  
Zürich, Switzerland  
daniele.pighin@gmail.com

**Carlo Strapparava**

FBK-irst  
Trento, Italy  
strappa@fbk.eu

## Abstract

We present BRAINSUP, an extensible framework for the generation of creative sentences in which users are able to force several words to appear in the sentences and to control the generation process across several semantic dimensions, namely *emotions*, *colors*, *domain relatedness* and *phonetic properties*. We evaluate its performance on a creative sentence generation task, showing its capability of generating well-formed, catchy and effective sentences that have all the good qualities of slogans produced by human copywriters.

## 1 Introduction

A variety of real-world scenarios involve talented and knowledgeable people in a time-consuming process to write creative, original sentences generated according to well-defined requisites. For instance, to advertise a new product it could be desirable to have its name appearing in a punchy sentence together with some keywords relevant for marketing, e.g. “fresh”, or “thirst” for the advertisement of a drink. Besides, it could be interesting to characterize the sentence with respect to a specific color, like “blue” to convey the idea of freshness, or to a color more related to the brand of the company, e.g. “red” for a new *Ferrari*. Moreover, making the slogan evoke “joy” or “satisfaction” could make the advertisement even more catchy for customers. On the other hand, there are many examples of provocative slogans in which copywriters try to impress their readers by suscitating strong negative feelings, as in the case of anti-smoke campaigns (e.g., “there are cooler ways to die than smoking” or “cancer cures smoking”), or the famous beer motto “*Guinness* is not good for

you”. As another scenario, creative sentence generation is also a useful teaching device. For example, the *keyword* or *linkword* method used for second language learning links the translation of a foreign (*target*) word to one or more keywords in the native language which are phonologically or lexically similar to the target word (Sagarra and Alba, 2006). To illustrate, for teaching the Italian word “tenda”, which means “curtain” in English, the learners are asked to imagine “rubbing a *tender* part of their leg with a *curtain*”. These words should co-occur in the same sentence, but constructing such sentences by hand can be a difficult and very time-consuming process. Özbal and Strapparava (2011), who attempted to automate the process, conclude that the inability to retrieve from the web a good sentence for all cases is a major bottleneck.

Although state of the art computational models of creativity often produce remarkable results, e.g., Manurung et al. (2008), Greene et al. (2010), Guerini et al. (2011), Colton et al. (2012) just to name a few, to our best knowledge there is no attempt to develop an unified framework for the generation of creative sentences in which users can control all the variables involved in the creative process to achieve the desired effect.

In this paper, we advocate the use of syntactic information to generate creative utterances by describing a methodology that accounts for lexical and phonetic constraints and multiple semantic dimensions at the same time. We present BRAINSUP, an extensible framework for creative sentence generation in which users can control all the parameters of the creative process, thus generating sentences that can be used for practical applications. First, users can define a set of keywords which must appear in the final sentence. Second, they can slant the output towards a spe-

Domain	Keywords	BRAINSUP output examples
coffee	waking, cup	Between waking and doing there is a wondrous cup.
coke	drink, exhaustion	The physical exhaustion wants the dark drink.
health	day, juice, sunshine	With juice and cereal the normal day becomes a summer sunshine.
beauty	kiss, lips	Passionate kiss, perfect lips. – Lips and eyes want the kiss.
mascara	drama, lash	Lash your drama to the stage. – A mighty drama, a biting lash.
pickle	crunch, bite	Crunch your bite to the top. – Crunch of a savage byte. – A large byte may crunch a little attention.
soap	skin, love, touch	A touch of love is worth a fortune of skin. – The touch of froth is the skin of love. – A skin of water is worth a touch of love.

Table 1: A selection of sentences automatically generated by BRAINSUP for specific domains.

cific *emotion*, *color* or *domain*. At the same time, they can require a sentence to include desired phonetic properties, such as *rhymes*, *alliteration* or *plosives*. The combination of these features allows for the generation of potentially catchy and memorable sentences by establishing connections between linguistic, emotional (LaBar and Cabeza, 2006), echoic and visual (Borman et al., 2005) memory, as exemplified by the system outputs showcased in Table 1. Other creative dimensions can easily be plugged in, due to the inherently modular structure of the system.

BRAINSUP supports the creative process by greedily exploring a huge solution space to produce completely novel utterances responding to user requisites. It exploits syntactic constraints to dramatically cut the size of the search space, thus making it possible to focus on the creative aspects of sentence generation.

## 2 Related work

Research in creative language generation has bloomed in recent years. In this section, we provide a necessarily succinct overview of a selection of the studies that most heavily inspired and influenced the development of BRAINSUP.

Humor generators are a notable class of systems exploring new venues in computational creativity (Binsted and Ritchie, 1997; McKay, 2002; Manurung et al., 2008). Valitutti et al. (2009) present an interactive system which generates humorous puns obtained through variation of famil-

iar expressions with word substitution. The variation takes place considering the phonetic distance and semantic constraints such as semantic similarity, semantic domain opposition and affective polarity difference. Possibly closer to slogan generation, Guerini et al. (2011) slant existing textual expressions to obtain more positively or negatively valenced versions using WordNet (Miller, 1995) semantic relations and SentiWordNet (Esuli and Sebastiani, 2006) annotations. Stock and Straparava (2006) generate acronyms based on lexical substitution via semantic field opposition, rhyme, rhythm and semantic relations. The model is limited to the generation of noun phrases.

Poetry generation systems face similar challenges to BRAINSUP as they struggle to combine semantic, lexical and phonetic features in a unified framework. Greene et al. (2010) describe a model for poetry generation in which users can control meter and rhyme scheme. Generation is modeled as a cascade of weighted Finite State Transducers that only accept strings conforming to the desired rhyming scheme. Toivanen et al. (2012) attempt to generate novel poems by replacing words in existing poetry with morphologically compatible words that are semantically related to a target domain. Content control and the inclusion of phonetic features are left as future work and syntactic information is not taken into account. The Electronic Text Composition project<sup>1</sup> is a corpus based approach to poetry generation which recursively combines automatically generated linguistic constituents into grammatical sentences. Colton et al. (2012) propose another data-driven approach to poetry generation based on simile transformation. The mood and theme of the poems are influenced by daily news. Constraints about phonetic properties of the selected words or their frequencies can be enforced during retrieval. Unlike these examples, BRAINSUP makes heavy use of syntactic information to enforce well-formed sentences and to constraint the search for a solution, and provides an extensible framework in which various forms of linguistic creativity can easily be incorporated.

Several slogan generators are available on the web<sup>2</sup>, but their capabilities are very limited as they can only replace single words or word sequences within existing slogan. This often results in syntactically incorrect outputs. Furthermore, they do not allow for other forms of user control.

<sup>1</sup><http://slought.org/content/11199>

<sup>2</sup>E.g.: <http://www.procato.com/slogan+generator>, <http://www.sloganizer.net/en/>, <http://www.sloganmania.com/index.htm>.

### 3 Architecture of BRAINSUP

To effectively support the creative process with useful suggestions, we must be able to generate sentences conforming to the user needs. First of all, users can select the *target words* that need to appear in the sentence. In the context of second language learning, these might be the words that a learner must associate in order to expand her vocabulary. For slogan generation, the target words could be the key features of a product, or target-defining keywords that copywriters want to explicitly mention. On top of that, a user can characterize the generated sentences according to several dimensions, namely: 1) a specific semantic domain, e.g.: “sports” or “blankets”; 2) a specific emotion, e.g., “joy”, “anger” or just “negative”; 3) a specific color, e.g., “red” or “blue”; 4) a combination of phonetic properties of the words that will appear in the sentence, i.e., rhymes, alliterations and plosives. More formally, the *user input* is a tuple:  $U = \langle \mathbf{t}, \mathbf{d}, c, e, p, \mathbf{w} \rangle$ , where  $\mathbf{t}$  is the set of *target words*,  $\mathbf{d}$  is a set of words defining the target *domain*,  $c$  and  $p$  are, respectively, the *color* and the *emotion* towards which the user wants to slant the sentence,  $p$  represents the desired phonetic features, and  $\mathbf{w}$  is a set of *weights* that control the influence of each dimension on the generative process, as detailed in Section 3.3. For target and domain words, users can explicitly select one or more POSes to be considered, e.g., “drink/verb” or “drink/verb,noun”.

The sentence generation process is based on morpho-syntactic *patterns* which we automatically discover from a corpus of dependency parsed sentences  $\mathcal{P}$ . These patterns represent very general skeletons of well-formed sentences that we employ to generate creative sentences by only focusing on the lexical aspects of the process. Candidate fillers for each empty position (*slot*) in the patterns are chosen according to the lexical and syntactic constraints enforced by the dependency relations in the patterns. These constraints are learned from relation-head-modifier co-occurrence counts estimated from a dependency treebank  $\mathcal{L}$ . A beam search in the space of all possible lexicalizations of a syntactic pattern promotes the words with the highest likelihood of satisfying the user specification.

Algorithm 1 provides a high-level description of the creative sentence generation process. Here,  $\Theta$  is a set of meta-parameters that affect search complexity and running time of the algorithm, such as the minimum/maximum number of solutions to

---

**Algorithm 1** SentenceGeneration( $U, \Theta, \mathcal{P}, \mathcal{L}$ ):  $U$  is the user specification,  $\Theta$  is a set of meta-parameters;  $\mathcal{P}$  and  $\mathcal{L}$  are two dependency treebanks.

---

```

 $\mathcal{O} \leftarrow \emptyset$ 
for all  $p \in \text{CompatiblePatterns}_{\Theta}(U, \mathcal{P})$  do
  while NotEnoughSolutions $_{\Theta}(\mathcal{O})$  do
     $\mathcal{O} \leftarrow \mathcal{O} \cup \text{FillInPattern}_{\Theta}(U, p, \mathcal{L})$ 
  return SelectBestSolutions $_{\Theta}(\mathcal{O})$ 

```

---

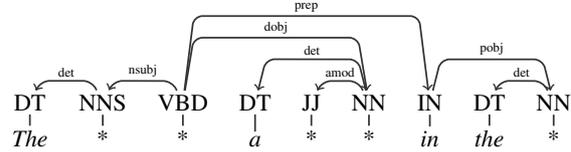


Figure 1: Example of a syntactic *pattern*. A “\*” represents an empty *slot* to be filled with a *filler*.

be generated, the maximum number of patterns to consider, or the maximum size of the generated sentences. CompatiblePatterns( $\cdot$ ) finds the most frequent syntactic patterns in  $\mathcal{P}$  that are compatible with the user specification, as explained in Section 3.1; FillInPattern( $\cdot$ ) carries out the beam search, and returns the best solutions generated for each pattern  $p$  given  $U$ . The algorithm terminates when at least a minimum number of solutions have been generated, or when all the compatible patterns have been exhausted. Finally, only the best among the generated solutions are shown to the user. More details about the search in the solution space are provided in Section 3.2.

#### 3.1 Pattern selection

We generate creative sentences starting from morpho-syntactic *patterns* which have been automatically learned from a large corpus  $\mathcal{P}$ . The choice of the corpus from which the patterns are extracted constitutes the first element of the creative sentence generation process, as different choices will generate sentences with different styles. For example, a corpus of slogans or punchlines can result in short, catchy and memorable sentences, whereas a corpus of simplified English would be a better choice to learn a second language or to address low reading level audiences.

A pattern is the syntactic skeleton of a class of sentences observed in  $\mathcal{P}$ . Within a pattern, a second element of creativity involves the selection of original combinations of words (*fillers*) that do not violate the grammaticality of the sentence. The patterns that we employ are automatic dependency trees from which all content-words have been removed, as exemplified in Figure 1. After selecting the target corpus, we parse all the sentences with the Stanford Parser (Klein and Man-

ning, 2003) and produce the patterns by stripping away all content words from the parses. Then, for each pattern we count how many times it has been observed in the corpus. Additionally, we keep track of what kind of empty *slots*, i.e., empty positions, are available in each pattern. For example, the pattern in Figure 1 can accommodate up to two singular nouns (NN), one plural noun (NNS), one adjective (JJ) and one verb in the past tense (VBD). This information is needed to select the patterns which are *compatible* with the target words  $\mathbf{t}$  in the user specification  $U$ . For example, this pattern is not compatible with  $\mathbf{t} = [\text{heading}/\text{VBG}, \text{edge}/\text{NN}]$  as the pattern does not have an empty slot for a gerundive verb, while it satisfies  $\mathbf{t} = [\text{heading}/\text{NN}, \text{edge}/\text{NN}]$  as it can accommodate the two singular nouns. While retrieving patterns, we also need to enforce that a pattern be not completely filled just by adding the target words  $\mathbf{t}$ , as under these conditions there would be no room to achieve any kind of creative effect. Therefore, we also require that the patterns retrieved by `CompatiblePatterns( $\cdot$ )` have more empty slots than the size of  $\mathbf{t}$ . The minimum and maximum number of excess slots in the pattern are two other meta-parameters controlled by  $\Theta$ . `CompatiblePatterns( $\cdot$ )` returns compatible patterns ordered by their frequency, i.e. when generating solutions the first patterns that are explored are the most frequently observed ones. In this way, we achieve the following two objectives: 1) we compensate for the unavoidable errors introduced by the automatic parser, as frequently observed parses are less likely to be the result of an erroneous interpretation of a sentence; and 2) we generate sentences that are most likely to be catchy and memorable, being based on syntactic constructs that are used more frequently. To avoid always selecting the same patterns for the same kinds of inputs, we add a small random component (also controlled by  $\Theta$ ) to the pattern sorting algorithm, thus allowing for sentences to be generated also from non-top ranked patterns.

### 3.2 Searching the solution space

With the compatible patterns selected, we can initiate a beam search in the space of all possible lexicalizations of the patterns, i.e., the space of all sentences that can be generated by respecting the syntactic constraints encoded by each pattern. The process starts with a syntactic pattern  $p$  containing only stop words, syntactic relations and morphologic constraints (i.e., part-of-speech

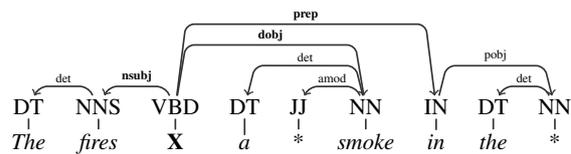


Figure 2: A partially lexicalized sentence with a highlighted empty slot marked with  $X$ . The relevant dependencies to fill in the slot are shown in boldface.

tags) for the empty slots. The search advances towards a complete solution by selecting an empty slot to fill and trying to place candidate fillers in the selected position. Each *partially lexicalized solution* is scored by a battery of scoring functions that compete to generate creative sentences respecting the user specification  $U$ , as explained in Section 3.3. The most promising solutions are extended by filling another slot, until completely lexicalized sentences, i.e., sentences without empty slots, are generated.

To limit the number of words that can occupy a given position in a sentence, we define a set of operators that return a list of candidate fillers for a slot solely based on syntactic clues. To achieve that, we analyze a large corpus of parsed sentences  $\mathcal{L}^3$  and store counts of observed *head-relation-modifier* ( $\langle h, r, m \rangle$ ) dependency relations. Let  $\tau_r(h)$  be an operator that, when applied to a head word  $h$  in a relation  $r$ , returns the set of words in  $\mathcal{L}$  which have been observed as modifiers for  $h$  in  $r$  with a specific POS. To simplify the notation, we assume that the relation  $r$  also carries along the POS of the head and modifier slots. As an example, with respect to the tree depicted in Figure 2,  $\tau_{\text{amod}}(\text{smoke})$  would return all the words with POS equal to “JJ” that have been observed as adjective modifiers for the singular noun “smoke”. We will refer to  $\tau_r(\cdot)$  as the *dependency operator* for  $r$ . For every  $\tau_r(\cdot)$ , we also define an *inverse dependency operator*  $\tau_r^{-1}(\cdot)$ , which returns the list of the possible heads in  $r$  when applied to a modifier word  $m$ . For instance, with respect to Figure 2,  $\tau_{\text{nsubj}}^{-1}(\text{fires})$  would return the set of verbs in the past tense of which “fires” as a plural noun can be a subject.

While filling in a given slot  $X$ , the dependency operators can be combined to obtain a list of words which are likely to occupy that position given the syntactic constraints induced by the structure of the pattern. Let  $W = \cup_i \{w_i\}$  be the set of words which are directly connected to the empty slot by

<sup>3</sup>Distinct from the corpus used for pattern selection,  $\mathcal{P}$ .

a dependency relation. Each word  $w_i$  implies a constraint that candidate fillers for  $X$  must satisfy. If  $w_i$  is the head of  $X$ , then a direct operator is used to retrieve a list of fillers that satisfy the  $i^{\text{th}}$  constraint. Conversely, if  $w_i$  is a modifier of  $X$ , an inverse operator is employed.

As an example, let us consider the partially completed sentence shown in Figure 2 having an empty slot marked with  $X$ . Here, the word “smoke” is a modifier for  $X$ , to which it is connected by a *doobj* relation. Therefore, we can exploit  $\tau_{\text{doobj}}^{-1}(\text{smoke})$  to obtain a ranked list of words that can occupy  $X$  according to this constraint. Similarly, the  $\tau_{\text{nsubj}}^{-1}(\text{fires})$  operator can be used to retrieve a list of verbs in the past tense that accept “fires” as *nsubj* modifier. Finally  $\tau_{\text{prep}}^{-1}(\text{in})$  can further restrict our options to verbs that accepts complements introduced by the preposition “in”. For example, the words “generated”, “produced”, “caused” or “formed” would be good candidates to fill in the slot considering all the previous constraints. More formally, we can define the set of candidate fillers for a slot  $X$ ,  $\mathcal{C}_X$ , as:  $\mathcal{C}_X = \tau_{\text{rh}_X}^{-1}(h_X) \cap (\bigcap_{w_i|w_i \in M_X} \tau_{r_{w_i,X}}(w_i))$ , where  $r_{w_i,X}$  is the type of relation between  $w_i$  and  $X$ ,  $M_X$  is the set of modifiers of  $X$  and  $h_X$  is the syntactic head of  $X$ .<sup>4</sup>

Concerning the order in which slots are filled, we start from those that have the highest number of dependencies (both head or modifiers) that have been already instantiated in the sentence, i.e., we start from the slots that are connected to the highest number of non-empty slots. In doing so we maximize the constraints that we can rely on when inserting a new word, and eventually generate more reliable outputs.

### 3.3 Filler selection and solution scoring

We have devised a set of *feature functions* that account for different aspects of the creative sentence generation process. By changing the weight  $\mathbf{w}$  of the feature functions in  $U$ , users can control the extent to which each creativity component will affect the sentence generation process, and tune the output of the system to better match their needs. As explained in the remainder of this section, feature functions are responsible for ranking the candidate slot fillers to be used during sentence generation and for selecting the best solutions to be

<sup>4</sup>An empty slot does not generate constraints for  $X$ . In addition, there might be cases in which it is not possible to find a filler that satisfies all the constraints at the same time. In such cases, all the fillers that satisfy the maximum number of constraints are considered.

---

**Algorithm 2** RankCandidates( $U, \mathbf{f}, c_1, c_2, s, X$ ):  $c_1$  and  $c_2$  are two candidate fillers for the slot  $X$  in the sentence  $s = [s_0, \dots, s_n]$ ;  $\mathbf{f}$  is the set of feature functions;  $U$  is the user specification.

---

```

 $\mathbf{s}^{c_1} \leftarrow \mathbf{s}, \mathbf{s}^{c_2} \leftarrow \mathbf{s}, \mathbf{s}^{c_1}[X] \leftarrow c_1, \mathbf{s}^{c_2}[X] \leftarrow c_2$ 
for all  $\mathbf{f} \in \text{SortFeatureFunctions}_{\Theta}(U, \mathbf{f})$  do
  if  $\mathbf{f}(\mathbf{s}^{c_1}, U) > \mathbf{f}(\mathbf{s}^{c_2}, U)$  then return  $c_1 \succ c_2$ 
  else if  $\mathbf{f}(\mathbf{s}^{c_1}, U) < \mathbf{f}(\mathbf{s}^{c_2}, U)$  then
    return  $c_1 \prec c_2$ 
return  $c_1 \equiv c_2$ 

```

---

shown to the users.

Algorithm 2 details the process of ranking candidate fillers. To compare two candidates  $c_1$  and  $c_2$  for the slot  $X$  in the sentence  $s$ , we first generate two sentences  $\mathbf{s}^{c_1}$  and  $\mathbf{s}^{c_2}$  in which the empty slot  $X$  is occupied by  $c_1$  and  $c_2$ , respectively. Then, we sort the feature functions based on their weights in descending order, and in turn we apply them to score the two sentences. As soon as we find a scorer for which one sentence is better than the other, we can take a decision about the ranking of the fillers. This approach makes it possible to establish a strict order of precedence among feature functions and to select fillers that have a highest chance of maximizing the user satisfaction.

Concerning the scoring of partial solutions and complete sentences, we adopt a simple linear combination of scoring functions. Let  $\mathbf{s}$  be a (partial) sentence,  $\mathbf{f} = [f_0, \dots, f_k]$  be the vector of scoring functions and  $\mathbf{w} = [w_0, \dots, w_k]$  the associated vector of weights in  $U$ . The overall score of  $\mathbf{s}$  is calculated as  $\text{score}(\mathbf{s}, U) = \sum_{i=0}^k w_i f_i(\mathbf{s}, U)$ . Solutions that do not contain all the required target words are discarded and not shown to the user.

Currently, the model employs the following 12 feature functions:

**Chromatic and emotional connotation.** The chromatic connotation of a sentence  $\mathbf{s} = [s_0, \dots, s_n]$  is computed as  $\mathbf{f}(\mathbf{s}, U) = \sum_{s_i} (\text{sim}(s_i, c) - \sum_{c_j \neq c} \text{sim}(s_i, c_j))$ , where  $c$  is the user selected target color and  $\text{sim}(s_i, c_j)$  is the degree of association between the word  $s_i$  and the color  $c_j$  as calculated by Mohammad (2011). All the words in the sentence which have an association with the target color  $c$  give a positive contribution, while those that are associated with a color  $c_i \neq c$  contribute negatively. Emotional connotation works exactly in the same way, but in this case word-emotion associations are taken from (Mohammad and Turney, 2010).

**Domain relatedness.** This feature function uses an LSA (Deerwester et al., 1990) vector space

model to measure the similarity between the words in the sentence and the target domain  $\mathbf{d}$  specified by the user. It is calculated as:  $f(\mathbf{s}, U) = \frac{\sum_{d_i} v(d_i) \cdot \sum_{s_i} v(s_i)}{\|\sum_{d_i} v(d_i)\| \cdot \|\sum_{s_i} v(s_i)\|}$  where  $v(\cdot)$  returns the representation of a word in the vector space.

**Semantic cohesion.** This feature behaves exactly like domain relatedness, with the only difference that it measures the similarity between the words in the sentence and the target words  $\mathbf{t}$ .

**Target-words scorer.** This feature function simply counts what fraction of the target words  $\mathbf{t}$  is present in a partial solution:  $f(\mathbf{s}, U) = (\sum_{s_i | s_i \in \mathbf{t}} 1) / |\mathbf{t}|$ . The target word scorer takes care of enforcing the presence of the target words in the sentences. Letting beam search find the best placement for the target words comes at no extra cost and results in a simple and elegant model.

**Phonetic features (plosives, alliteration and rhyme).** All the phonetic features are based on the phonetic representation of English words of the Carnegie Mellon University pronouncing dictionary (Lenzo, 1998). The plosives feature is calculated as the ratio between the number of plosive sounds in a sentence and the overall number of phonemes. For the alliteration scorer, we store the phonetic representation of each word in  $\mathbf{s}$  in a *trie* (i.e., *prefix tree*), and count how many times each node  $n_i$  of the trie (corresponding to a phoneme) is traversed. Let  $c_i$  be the value of the counts for  $n_i$ . The alliteration score is then calculated as  $f(\mathbf{s}, U) = (\sum_{i | c_i > 1} c_i) / \sum_i c_i$ . More simply put, we count how many of the phonetic prefixes of the words in the sentence are repeated, and then we normalize this value by the total number of phonemes in  $\mathbf{s}$ . The rhyme feature works exactly in the same way, with the only difference that we invert the phonetic representation of each word before adding it to the TRIE. Thus, we give higher scores to sentences in which several words share the same phonetic ending.

**Variety scorer.** This feature function promotes sentences that contain as many different words as possible. It is calculated as the number of distinct words in the sentence over the size of the sentence.

**Unusual-words scorer.** To increase the ability of the model to generate sentences containing non-trivial word associations, we may want to prefer solutions in which relatively uncommon words are employed. Inversely, we may want to lower lex-

ical complexity to generate sentences more appropriate for certain education or reading levels. We define  $c_i$  as the number of times each word  $s_i \in \mathbf{s}$  is observed in a corpus  $\mathcal{V}$ . Accordingly, the value of this feature is calculated as:  $f(\mathbf{s}, U) = (1/|\mathbf{s}|)(\sum_{s_i} 1/c_i)$ .

**N-gram likelihood.** This is simply the likelihood of a sentence estimated by an  $n$ -gram language model, to enforce the generation of well-formed word sequences. When a solution is not complete, in the computation we include only the sequences of contiguous words (i.e., not interrupted by empty slots) having length greater than or equal to the order of the  $n$ -gram model.

**Dependency likelihood.** This feature is related to the dependency operators introduced in Section 3.2 and it enforces sentences in which dependency chains are well formed. We estimate the probability of a modifier word  $m$  and its head  $h$  to be in the relation  $r$  as  $p_r(h, m) = c_r(h, m) / (\sum_{h_i} \sum_{m_i} c_r(h_i, m_i))$ , where  $c_r(\cdot)$  is the number of times that  $m$  depends on  $h$  in the dependency treebank  $\mathcal{L}$  and  $h_i, m_i$  are all the head/modifier pairs observed in  $\mathcal{L}$ . The dependency-likelihood of a sentence  $\mathbf{s}$  can then be calculated as  $f(\mathbf{s}, U) = \exp(\sum_{(h,m,r) \in r(\mathbf{s})} \log p_r(h, m))$ ,  $r(\mathbf{s})$  being the set of dependency relations in  $\mathbf{s}$ .

## 4 Evaluation

We evaluated our model on a creative sentence generation task. The objective of the evaluation is twofold: we wanted to demonstrate 1) the effectiveness of our approach for creative sentence generation, in general, and 2) the potential of BRAINSUP to support the brainstorming process behind slogan generation. To this end, the annotation template included one question asking the annotators to rate the quality of the generated sentences as slogans.

Five experienced annotators were asked to rate 432 creative sentences according to the following criteria, namely: 1) *Catchiness*: is the sentence attractive, catchy or memorable? [Yes/No] 2) *Humor*: is the sentence witty or humorous? [Yes/No]; 3) *Relatedness*: is the sentence semantically related to the target domain? [Yes/No]; 4) *Correctness*: is the sentence grammatically correct? [Ungrammatical/Slightly disfluent/Fluent]; 5) *Success*: could the sentence be a good slogan for the target domain? [As it is/With minor editing/No]. In these last two cases, the annotators

were instructed to select the middle option only in cases where the gap with a correct/successful sentence could be filled just by performing minor editing. The annotation form had no default values, and the annotators did not know how the evaluated sentences were generated, or whether they were the outcome of one or more systems.

We started by collecting slogans from an online repository of slogans<sup>5</sup>. Then, we randomly selected a subset of these slogans and for each of them we generated an input specification  $U$  for the system. We used the commercial domain of the advertised product as the target domain  $\mathbf{d}$ . Two or three content words appearing in each slogan were randomly selected as the target words  $\mathbf{t}$ . We did so to simulate the brainstorming phase behind the slogan generation process, where copywriters start with a set of relevant keywords to come up with a catchy slogan. In all cases, we set the target emotion to “positive” as we could not establish a generally valid criteria to associate a specific emotion to a product. Concerning chromatic slanting, for target domains having a strong chromatic correlation we allowed the system to slant the generated sentences accordingly. In the other cases, a random color association was selected. In this manner, we produced 10 tuples  $\langle \mathbf{t}, \mathbf{d}, c, e, p \rangle$ . Then, from each tuple we produced 5 complete user specifications by enabling or disabling different feature function combinations<sup>6</sup>. The four combinations of features are: **base**: Target-word scorer + N-gram likelihood + Dependency likelihood + Variety scorer + Unusual-words scorer + Semantic cohesion; **base+D**: all the scorers in *base* + Domain relatedness; **base+D+C**: all the scorers in *base+D* + Chromatic connotation; **base+D+E**: all the scorers in *base+D* + Emotional connotation; **base+D+P**: all the scorers in *base+D* + Phonetic features. For each of the resulting 50 input configurations, we generated up to 10 creative sentences. As the system could not generate exactly 10 solutions in all the cases, we ended up with a set of 432 items to annotate. The weights of the feature functions were set heuristically, due to the lack of an annotated dataset suitable to learn an opti-

<sup>5</sup>[http://www.tvacres.com/advertising\\_slogans.htm](http://www.tvacres.com/advertising_slogans.htm)

<sup>6</sup>An alternative strategy to keep the annotation effort under control would have been to generate fewer sentences from a larger number of inputs. We adopted the former setting since we regarded it as more similar to a brainstorming session, where the system proposes different alternatives to inspire human operators. Forcing BRAINSUP to only output one or two sentences would have limited its ability to explore and suggest potentially valuable outputs.

MC	Cat.	Hum.	Corr.	Rel.	Succ.	RND <sub>2</sub>	RND <sub>3</sub>
2	-	-	16.67	-	22.22	-	37.04
3	47.45	39.58	43.52	13.66	44.21	62.50	49.38
4	33.10	37.73	32.18	21.99	22.22	31.25	12.35
5	19.44	22.69	07.64	64.35	11.34	06.25	01.23

Table 2: Majority classes (%) for the five dimensions of the annotation.

mal weight configuration. We started by assigning the highest weight to the *Target Word* scorer (i.e., 1.0), followed by the *Variety* and *Unusual Word* scorers (0.99), the *Phonetic Features*, *Chromatic/Emotional Connotation* and *Semantic Cohesion* scorers (0.98) and finally the *Domain*, *N-gram* and *Dependency Likelihood* scorers (0.97). These settings allow us to enforce an order of precedence among the scorers during slot-filling, while giving them virtually equal relevance for solution ranking.

As discussed in Section 3 we use two different treebanks to learn the syntactic patterns ( $\mathcal{P}$ ) and the dependency operators ( $\mathcal{L}$ ). For these experiments, patterns were learned from a corpus of 16,000 proverbs (Mihalcea and Strapparava, 2006), which offers a good selection of short sentences with a good potential to be used for slogan generation. This choice seemed to be a good compromise as, to our best knowledge, there is no published slogan dataset with an adequate size. Besides, using existing slogans might have legal implications that we might not be aware of. Dependency operators were learned by dependency parsing the British National Corpus<sup>7</sup>. To reduce the amount of noise introduced by the automatic parses, we only considered sentences having less than 20 words. Furthermore, we only considered sentences in which all the content words are listed in WordNet (Miller, 1995) with the observed part of speech.<sup>8</sup> The LSA space used for the semantic feature functions was also learned on BNC data, but in this case no filtering was applied.

## 4.1 Results

To measure the agreement among the annotators, similarly to Mohammad (2011) and Ozbal and Strapparava (2012) we calculated the majority class for each dimension of the annotation task. A

<sup>7</sup><http://www.natcorp.ox.ac.uk/>

<sup>8</sup>Since the CMU pronouncing dictionary used by the phonetic scorers is based on the American pronunciation of words, we actually pre-processed the whole BNC by replacing all British-English words with their American-English counterparts. To this end, we used the mapping available at <http://wordlist.sourceforge.net/>.

	Cat.	Rel.	Hum.	Succ.	Corr.
Yes	67.59	93.98	12.73	32.41	64.35
Partly	-	-	-	23.15	31.71
No	32.41	06.02	87.27	44.44	03.94

Table 3: Majority decisions (%) for each annotation dimension.

majority class greater than or equal to 3 means that the absolute majority of the 5 annotators agreed on the same decision<sup>9</sup>. Table 2 shows the observed agreement for each dimension. The column labeled  $RND_2$  ( $RND_3$ ) shows the random agreement for a given number of annotators and a binary (ternary) decision. For example, all five annotators ( $MC=5$ ) agreed on the annotation of the catchiness of the slogans in 19.44% of the cases. The random chance of agreement for 5 annotators on the binary decision problem is 6.25%. The figures for  $MC \geq 4$  are generally high, confirming a good agreement among the annotators. The agreement on the relatedness of the slogans is especially high, with all 5 annotators taking the same decision in almost two cases out of three, i.e., 64.35%.

Table 3 lists the distribution of answers for each dimension in the cases where a decision can be taken by majority vote. The generated slogans are found to be catchy in more than 2/3 of the cases, (i.e., 67.59%), completely successful in 1/3 of the cases (32.41%) and completely correct in 2/3 of the cases (64.35%). These figures demonstrate that BRAINSUP is very effective in generating grammatical utterances that have all the appealing properties of a successful slogan. As for humor, the sentences are found to have this property in only 12.73% of cases. Even though the figure is not very high, we should also consider that BRAINSUP is not explicitly trying to generate amusing utterances. Concerning success, we should point out that in 23.15% of the cases the annotators have found that the generated slogans have the potential to be turned into successful ones only with minor editing. This is a very important piece of result, as it corroborates our claim that BRAINSUP can indeed be a valuable tool for copywriting, even when it does not manage to output a perfectly good sentence. Similar conclusions can be drawn concerning the correctness of the output, as in almost one third of the cases the slogans are

<sup>9</sup>For the binary decisions (i.e., catchiness, relatedness and humor), at least 3 annotators out of 5 must necessarily agree on the same option.

only affected by minor disfluencies.

The relatedness figure is especially high, as in almost 94% of the cases the majority of annotators found the slogans to be pertinent to the target domain. This result is not surprising, as all the slogans are generated by considering keywords that already exist in real slogans for the same domain. Anyhow, this is exactly the kind of setting in which we expect BRAINSUP to be employed, i.e., to support creative sentence generation starting from a good set of relevant keywords. Nonetheless, it is very encouraging to observe that the generation process does not deteriorate the positive impact of the input keywords.

We would also like to mention that in 63 cases (14.58%) the majority of the annotators have labeled the slogans favorably across all 5 dimensions. The examples listed in Table 1 are selected from this set. It is interesting to observe how the word associations established by BRAINSUP can result in pertinent yet unintentional rhetorical devices such as metaphors (“a summer sunshine”), puns (“lash your drama”) and personifications (“lips and eyes want”). Some examples show the effect of the phonetic features, e.g. plosives in “*passionate kiss, perfect lips*”, alliteration in “*the dark drink*” and rhyming in “lips and eyes want the kiss”. In some cases, the output of BRAINSUP seems to be governed by mysterious philosophical reasoning, as in the delicate examples generated for “soap”.

For comparison, Table 4 lists a selection of the examples that have been labeled as unsuccessful by the majority of raters. In some cases, BRAINSUP is improperly selecting attributes that highlight undesirable properties in the target domain, e.g., “A pleasant tasting, a *heady* wine”. To avoid similar errors, it would be necessary to reason about the valence of an attribute for a specific domain. In other cases, the  $N$ -gram and the Dependency Likelihood features may introduce phrases which are very cohesive but unrelated to the rest of the sentence, e.g., “Unscrupulous doctors smoke *armored units*”. Many of these errors could be solved by increasing the weight of the Semantic Cohesion and Domain Relatedness scorers. In other cases, such as “A sixth calorie may taste *an own good*” or “A *same sunshine* is fewer than a juice of day”, more sophisticated reasoning about syntactic and semantic relations in the output might be necessary in order to enforce the generation of sound and grammatical sentences.

We could not find a significant correlation be-

Domain	Keywords	BRAINSUP output examples
pleasure	wine, tasting	A pleasant tasting, a heady wine. – A fruity tasting may drink a sparkling wine.
healthy	day, juice, sunshine	Drink juice of your sunshine, and your weight will choose day of you. – A same sunshine is fewer than a juice of day.
cigarette	doctors, smoke	Unscrupulous doctors smoke armored units. – Doctors smoke no arrow.
mascara	drama, lash	The such drama is the lash.
soap	skin, love, touch	The touch of skin is the love of cacophony. – You love an own skin for a first touch.
coke	calorie, taste, good	A sixth calorie may taste an own good.
coffee	waking, cup	You cannot cup hands without waking some fats.

Table 4: Unsuccessful BRAINSUP outputs.

tween the input variables (e.g., presence or absence of phonetic features or chromatic slanting) and the outcome of the annotation, i.e. the system by and large produces correct, catchy, related and (at least potentially) successful outputs regardless of the specific input configurations. In this respect, it should be noted that we did not carry out any kind of optimization of the feature weights, which might be needed to obtain more heavily characterized sentences. Furthermore, to better appreciate the contribution of the individual features, comparative experiments in which the users evaluate the system before and after triggering a feature function might be necessary. Concerning the correlation among output dimensions, we only observed relatively high Spearman correlation between correctness and relatedness (0.65), and catchiness and success (0.68).

## 5 Conclusion

We have presented BRAINSUP, a novel system for creative sentence generation that allows users to control many aspects of the creativity process, from the presence of specific target words in the output, to the selection of a target domain, and to the injection of phonetic and semantic properties in the generated sentences. BRAINSUP makes heavy use of dependency parsed data and statistics collected from dependency treebanks to ensure the grammaticality of the generated sentences, and to trim the search space while seeking the sentences that maximize the user satisfaction.

The system has been designed as a supporting tool for a variety of real-world applications, from advertisement to entertainment and education, where at the very least it can be a valuable support for time-consuming and knowledge-intensive sentence generation needs. To demonstrate this point, we carried out an evaluation on a creative sentence generation benchmark showing that BRAINSUP can effectively produce catchy, memorable and successful sentences that have the potential to inspire the work of copywriters.

To our best knowledge, this is the first systematic attempt to build an extensible framework that allows for multi-dimensional creativity while at the same time relying on syntactic constraints to enforce grammaticality. In this regard, our approach is dual with respect to previous work based on lexical substitution, which suffers from limited expressivity and creativity latitude. In addition, by acquiring the lexicon and the sentence structure from two distinct corpora, we can guarantee that the sentences that we generate have never been observed. We believe that our contribution constitutes a valid starting point for other researchers to deal with unexplored dimensions of creativity.

As future work, we plan to use machine learning techniques to estimate optimal weights for the feature functions in different use cases. We would also like to consider syntactic clues while reasoning about semantic properties of the sentence, e.g., color and emotion associations, instead on relying solely on lexical semantics. Concerning the extension of the capabilities of BRAINSUP, we want to include common-sense knowledge and reasoning to profit from more sophisticated semantic relations and to inject humor on demand. Further tuning of BRAINSUP to build a dedicated system for slogan generation is also part of our future plans. After these improvements, we would like to conduct a more focused evaluation on slogan generation involving human copywriters and domain experts in an interactive setting.

We would like to conclude this paper with a pearl of BRAINSUP’s wisdom:

*It is wiser to believe in science  
than in everlasting love.*

## Acknowledgments

Gözde Özbal and Carlo Strapparava were partially supported by the PerTe project (Trento RISE).

## References

- Kim Binsted and Graeme Ritchie. 1997. Computational rules for generating punning riddles. *Humor - International Journal of Humor Research*, 10(1):25–76, January.
- Andy Borman, Rada Mihalcea, and Paul Tarau. 2005. Pic-net: Pictorial representations for illustrated semantic networks. In *Proceedings of the AAAI Spring Symposium on Knowledge Collection from Volunteer Contributors*.
- Simon Colton, Jacob Goodwin, and Tony Veale. 2012. Full-FACE Poetry Generation. In *Proceedings of the 3rd International Conference on Computational Creativity*, pages 95–102.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal Of The American Society for Information Science*, 41(6):391–407.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentimentnet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC'06)*, pages 417–422.
- Erica Greene, Tugba Bodrumlu, and Kevin Knight. 2010. Automatic analysis of rhythmic poetry with applications to generation and translation. In *EMNLP*, pages 524–533.
- Marco Guerini, Carlo Strapparava, and Oliviero Stock. 2011. Slanting existing text with valentino. In *Proceedings of the 16th international conference on Intelligent user interfaces, IUI '11*, pages 439–440, New York, NY, USA. ACM.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03*, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kevin S. LaBar and Roberto Cabeza. 2006. Cognitive neuroscience of emotional memory. *Nature reviews. Neuroscience*, 7(1):54–64, January.
- Kevin Lenzo. 1998. The cmu pronouncing dictionary. <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>.
- Ruli Manurung, Graeme Ritchie, Helen Pain, Analu Waller, Dave O'Mara, and Rolf Black. 2008. The Construction of a Pun Generator for Language Skills Development. *Applied Artificial Intelligence*, 22(9):841–869, October.
- J McKay. 2002. Generation of idiom-based witticisms to aid second language learning. In *Twente Workshop on Language Technology 20*, pages 70–74.
- R. Mihalcea and C. Strapparava. 2006. Learning to laugh (automatically): Computational models for humor recognition. *Journal of Computational Intelligence*, 22(2):126–142, May.
- George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41.
- Saif M. Mohammad and Peter D. Turney. 2010. Emotions evoked by common words and phrases: using mechanical turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text, CAAGET '10*, pages 26–34, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Saif Mohammad. 2011. Even the abstract have color: Consensus in word-colour associations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–373, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Gözde Özbal and Carlo Strapparava. 2011. Automated Memory Techniques for Vocabulary Acquisition in a Second Language. In Alexander Verbraeck, Markus Helfert, José Cordeiro, and Boris Shishkov, editors, *CSEU*, pages 79–87. SciTePress.
- Gozde Ozbal and Carlo Strapparava. 2012. A computational approach to the automation of creative naming. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 703–711, Jeju Island, Korea, July. Association for Computational Linguistics.
- N. Sagarra and M. Alba. 2006. The key is in the keyword: L2 vocabulary learning methods with beginning learners of spanish. *The Modern Language Journal*, 90(2):228–243.
- Oliviero Stock and Carlo Strapparava. 2006. Laughing with hahacronym, a computational humor system. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, pages 1675–1678. AAAI Press.
- J. M. Toivanen, H. Toivonen, A. Valitutti, and O. Gross. 2012. Corpus-based Generation of Content and Form in Poetry. In *International Conference on Computational Creativity*, pages 175–179.
- A. Valitutti, C. Strapparava, , and O. Stock. 2009. Graphlaugh: a tool for the interactive generation of humorous puns. In *Proceedings of ACII-2009, Third Conference on Affective Computing and Intelligent Interaction, Demo track*.

# Grammatical Error Correction Using Integer Linear Programming

**Yuanbin Wu**

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

**Hwee Tou Ng**

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

## Abstract

We propose a joint inference algorithm for grammatical error correction. Different from most previous work where different error types are corrected independently, our proposed inference process considers all possible errors in a unified framework. We use integer linear programming (ILP) to model the inference process, which can easily incorporate both the power of existing error classifiers and prior knowledge on grammatical error correction. Experimental results on the Helping Our Own shared task show that our method is competitive with state-of-the-art systems.

## 1 Introduction

Grammatical error correction is an important task of natural language processing (NLP). It has many potential applications and may help millions of people who learn English as a second language (ESL). As a research field, it faces the challenge of processing ungrammatical language, which is different from other NLP tasks. The task has received much attention in recent years, and was the focus of two shared tasks on grammatical error correction in 2011 and 2012 (Dale and Kilgarriff, 2011; Dale et al., 2012).

To detect and correct grammatical errors, two different approaches are typically used — knowledge engineering or machine learning. The first relies on handcrafting a set of rules. For example, the superlative adjective *best* is preceded by the article *the*. In contrast, the machine learning approach formulates the task as a classification problem based on learning from training data. For example, an article classifier takes a noun phrase (NP) as input and predicts its article using class labels *a/an*, *the*, or  $\epsilon$  (no article).

Both approaches have their advantages and disadvantages. One can readily handcraft a set of

rules to incorporate various prior knowledge from grammar books and dictionaries, but rules often have exceptions and it is difficult to build rules for all grammatical errors. On the other hand, the machine learning approach can learn from texts written by ESL learners where grammatical errors have been annotated. However, training data may be noisy and classifiers may need prior knowledge to guide their predictions.

Another consideration in grammatical error correction is how to deal with multiple errors in an input sentence. Most previous work deals with errors individually: different classifiers (or rules) are developed for different types of errors (article classifier, preposition classifier, etc). Classifiers are then deployed independently. An example is a pipeline system, where each classifier takes the output of the previous classifier as its input and proposes corrections of one error type.

One problem of this pipeline approach is that the relations between errors are ignored. For example, assume that an input sentence contains *a cats*. An article classifier may propose to delete *a*, while a noun number classifier may propose to change *cats* to *cat*. A pipeline approach will choose one of the two corrections based purely on which error classifier is applied first. Another problem is that when applying a classifier, the surrounding words in the context are assumed to be correct, which is not true if grammatical errors appear close to each other in a sentence.

In this paper, we formulate grammatical error correction as a task suited for joint inference. Given an input sentence, different types of errors are jointly corrected as follows. For every possible error correction, we assign a score which measures how grammatical the resulting sentence is if the correction is accepted. We then choose a set of corrections which will result in a corrected sentence that is judged to be the most grammatical.

The inference problem is solved by integer lin-

ear programming (ILP). Variables of ILP are indicators of possible grammatical error corrections, the objective function aims to select the best set of corrections, and the constraints help to enforce a valid and grammatical output. Furthermore, ILP not only provides a method to solve the inference problem, but also allows for a natural integration of grammatical constraints into a machine learning approach. We will show that ILP fully utilizes individual error classifiers, while prior knowledge on grammatical error correction can be easily expressed using linear constraints. We evaluate our proposed ILP approach on the test data from the Helping Our Own (HOO) 2011 shared task (Dale and Kilgarriff, 2011). Experimental results show that the ILP formulation is competitive with state-of-the-art grammatical error correction systems.

The remainder of this paper is organized as follows. Section 2 gives the related work. Section 3 introduces a basic ILP formulation. Sections 4 and 5 improve the basic ILP formulation with more constraints and second order variables, respectively. Section 6 presents the experimental results. Section 7 concludes the paper.

## 2 Related Work

The knowledge engineering approach has been used in early grammatical error correction systems (Murata and Nagao, 1993; Bond et al., 1995; Bond and Ikehara, 1996; Heine, 1998). However, as noted by (Han et al., 2006), rules usually have exceptions, and it is hard to utilize corpus statistics in handcrafted rules. As such, the machine learning approach has become the dominant approach in grammatical error correction.

Previous work in the machine learning approach typically formulates the task as a classification problem. Article and preposition errors are the two main research topics (Knight and Chander, 1994; Han et al., 2006; Tetreault and Chodorow, 2008; Dahlmeier and Ng, 2011). Features used in classification include surrounding words, part-of-speech tags, language model scores (Gamon, 2010), and parse tree structures (Tetreault et al., 2010). Learning algorithms used include maximum entropy (Han et al., 2006; Tetreault and Chodorow, 2008), averaged perceptron, naïve Bayes (Rozovskaya and Roth, 2011), etc. Besides article and preposition errors, verb form errors also attract some attention recently (Liu et al., 2010; Tajiri et al., 2012).

Several research efforts have started to deal with correcting different errors in an integrated manner (Gamon, 2011; Park and Levy, 2011; Dahlmeier and Ng, 2012a). Gamon (2011) uses a high-order sequential labeling model to detect various errors. Park and Levy (2011) models grammatical error correction using a noisy channel model, where a predefined generative model produces correct sentences and errors are added through a noise model. The work of (Dahlmeier and Ng, 2012a) is probably the closest to our current work. It uses a beam-search decoder, which iteratively corrects an input sentence to arrive at the best corrected output. The difference between their work and our ILP approach is that the beam-search decoder returns an approximate solution to the original inference problem, while ILP returns an exact solution to an approximate inference problem.

Integer linear programming has been successfully applied to many NLP tasks, such as dependency parsing (Riedel and Clarke, 2006; Martins et al., 2009), semantic role labeling (Punyakanok et al., 2005), and event extraction (Riedel and McCallum, 2011).

## 3 Inference with First Order Variables

The inference problem for grammatical error correction can be stated as follows: “*Given an input sentence, choose a set of corrections which results in the best output sentence.*” In this paper, this problem will be expressed and solved by integer linear programming (ILP).

To express an NLP task in the framework of ILP requires the following steps:

1. Encode the output space of the NLP task using integer variables;
2. Express the inference objective as a linear objective function; and
3. Introduce problem-specific constraints to refine the feasible output space.

In the following sections, we follow the above formulation. For the grammatical error correction task, the variables in ILP are indicators of the corrections that a word needs, the objective function measures how grammatical the whole sentence is if some corrections are accepted, and the constraints guarantee that the corrections do not conflict with each other.

### 3.1 First Order Variables

Given an input sentence, the main question that a grammatical error correction system needs to answer is: *What corrections at which positions?* For example, is it reasonable to change the word *cats* to *cat* in the sentence *A cats sat on the mat?* Given the corrections at various positions in a sentence, the system can readily come up with the corrected sentence. Thus, a natural way to encode the output space of grammatical error correction requires information about sentence position, error type (e.g., noun number error), and correction (e.g., *cat*).

Suppose  $s$  is an input sentence, and  $|s|$  is its length (i.e., the number of words in  $s$ ). Define *first order variables*:

$$Z_{l,p}^k \in \{0, 1\}, \quad (1)$$

where

$p \in \{1, 2, \dots, |s|\}$  is a position in a sentence,

$l \in L$  is an error type,

$k \in \{1, 2, \dots, C(l)\}$  is a correction of type  $l$ .

$L$ : the set of error types,

$C(l)$ : the number of corrections for error type  $l$ .

If  $Z_{l,p}^k = 1$ , the word at position  $p$  should be corrected to  $k$  that is of error type  $l$ . Otherwise, the word at position  $p$  is not applicable for this correction. Deletion of a word is represented as  $k = \epsilon$ . For example,  $Z_{\text{Art},1}^a = 1$  means that the article (Art) at position 1 of the sentence should be  $a$ . If  $Z_{\text{Art},1}^a = 0$ , then the article should not be  $a$ . Table 1 contains the error types handled in this work, their possible corrections and applicable positions in a sentence.

### 3.2 The Objective Function

The objective of the inference problem is to find the best output sentence. However, there are exponentially many different combinations of corrections, and it is not possible to consider all combinations. Therefore, instead of solving the original inference problem, we will solve an approximate inference problem by introducing the following decomposable assumption: *Measuring the output quality of multiple corrections can be decomposed into measuring the quality of the individual corrections.*

Let  $s'$  be the resulting sentence if the correction  $Z_{l,p}^k$  is accepted for  $s$ , or for simplicity denoting

it as  $s \xrightarrow{Z_{l,p}^k} s'$ . Let  $w_{l,p,k} \in \mathbb{R}$ , measure how grammatical  $s'$  is. Define the objective function as

$$\max \sum_{l,p,k} w_{l,p,k} Z_{l,p}^k.$$

This linear objective function aims to select a set of  $Z_{l,p}^k$ , such that the sum of their weights is the largest among all possible candidate corrections, which in turn gives the most grammatical sentence under the decomposable assumption.

Although the decomposable assumption is a strong assumption, it performs well in practice, and one can relax the assumption by using higher order variables (see Section 5).

For an individual correction  $Z_{l,p}^k$ , we measure the quality of  $s'$  based on three factors:

1. The language model score  $h(s', \text{LM})$  of  $s'$  based on a large web corpus;

2. The confidence scores  $f(s', t)$  of classifiers, where  $t \in E$  and  $E$  is the set of classifiers. For example, an article classifier trained on well-written documents will score every article in  $s'$ , and measure the quality of  $s'$  from the perspective of an article “expert”.

3. The disagreement scores  $g(s', t)$  of classifiers, where  $t \in E$ . A disagreement score measures how ungrammatical  $s'$  is from the perspective of a classifier. Take the article classifier as an example. For each article instance in  $s'$ , the classifier computes the difference between the maximum confidence score among all possible choices of articles, and the confidence score of the observed article. This difference represents the disagreement on the observed article by the article classifier or “expert”. Define the maximum difference over all article instances in  $s'$  to be the article classifier disagreement score of  $s'$ . In general, this score is large if the sentence  $s'$  is more ungrammatical.

The weight  $w_{l,p,k}$  is a combination of these scores:

$$w_{l,p,k} = \nu_{\text{LM}} h(s', \text{LM}) + \sum_{t \in E} \lambda_t f(s', t) + \sum_{t \in E} \mu_t g(s', t), \quad (2)$$

where  $\nu_{\text{LM}}$ ,  $\lambda_t$ , and  $\mu_t$  are the coefficients.

### 3.3 Constraints

An observation on the objective function is that it is possible, for example, to set  $Z_{\text{Art},p}^a = 1$  and

Type $l$	Correction $k$	$C(l)$	Applicable	Variables
article	a, the, $\epsilon$	3	article or NP	$Z_{\text{Art},p}^a, Z_{\text{Art},p}^{\text{the}}, Z_{\text{Art},p}^\epsilon$
preposition	on, at, in, ...	confusion set	preposition	$Z_{\text{Prep},p}^{\text{on}}, Z_{\text{Prep},p}^{\text{at}}, Z_{\text{Prep},p}^{\text{in}}, \dots$
noun number	singular, plural	2	noun	$Z_{\text{Noun},p}^{\text{singular}}, Z_{\text{Noun},p}^{\text{plural}}$
punctuation	punctuation symbols	candidates	determined by rules	$Z_{\text{Punct},p}^{\text{original}}, Z_{\text{Punct},p}^{\text{cand1}}, Z_{\text{Punct},p}^{\text{cand2}}, \dots$
spelling	correctly spelled words	candidates	determined by a spell checker	$Z_{\text{Spell},p}^{\text{original}}, Z_{\text{Spell},p}^{\text{cand1}}, Z_{\text{Spell},p}^{\text{cand2}}, \dots$

Table 1: Error types and corrections. The *Applicable* column indicates which parts of a sentence are applicable to an error type. In the first row,  $\epsilon$  means deleting an article.

$Z_{\text{Art},p}^{\text{the}} = 1$ , which means there are two corrections *a* and *the* for the same sentence position  $p$ , but obviously only one article is allowed.

A simple constraint to avoid these conflicts is

$$\sum_k Z_{l,p}^k = 1, \quad \forall \text{ applicable } l, p$$

It reads as follows: for each error type  $l$ , only one output  $k$  is allowed at any applicable position  $p$  (note that  $Z_{l,p}^k$  is a Boolean variable).

Putting the variables, objective function, and constraints together, the ILP problem with respect to first order variables is as follows:

$$\max \sum_{l,p,k} w_{l,p,k} Z_{l,p}^k \quad (3)$$

$$\text{s.t. } \sum_k Z_{l,p}^k = 1, \quad \forall \text{ applicable } l, p \quad (4)$$

$$Z_{l,p}^k \in \{0, 1\} \quad (5)$$

The ILP problem is solved using `lp_solve`<sup>1</sup>, an integer linear programming solver based on the revised simplex method and the branch-and-bound method for integers.

### 3.4 An Illustrating Example

To illustrate the ILP formulation, consider an example input sentence  $s$ :

$$\text{A cats sat on the mat .} \quad (6)$$

First, the constraint (4) at position 1 is:

$$Z_{\text{Art},1}^a + Z_{\text{Art},1}^{\text{the}} + Z_{\text{Art},1}^\epsilon = 1,$$

which means only one article in  $\{a, \text{the}, \epsilon\}$  is selected.

<sup>1</sup><http://lpsolve.sourceforge.net/>

Next, to compute  $w_{l,p,k}$ , we collect language model score and confidence scores from the article (ART), preposition (PREP), and noun number (NOUN) classifier, i.e.,  $E = \{\text{ART}, \text{PREP}, \text{NOUN}\}$ .

The weight for  $Z_{\text{Noun},2}^{\text{singular}}$  is:

$$w_{\text{Noun},2,\text{singular}} = \nu_{\text{LM}} h(s', \text{LM}) + \lambda_{\text{ART}} f(s', \text{ART}) + \lambda_{\text{PREP}} f(s', \text{PREP}) + \lambda_{\text{NOUN}} f(s', \text{NOUN}) + \mu_{\text{ART}} g(s', \text{ART}) + \mu_{\text{PREP}} g(s', \text{PREP}) + \mu_{\text{NOUN}} g(s', \text{NOUN}).$$

where  $s \xrightarrow{Z_{\text{Noun},2}^{\text{singular}}} s' = \text{A cat sat on the mat .}$

The confidence score  $f(s', t)$  of classifier  $t$  is the average of the confidence scores of  $t$  on the applicable instances in  $s'$ .

For example, there are two article instances in  $s'$ , located at position 1 and 5 respectively, hence,

$$\begin{aligned} f(s', \text{ART}) &= \frac{1}{2} (f(s'[1], 1, \text{ART}) + f(s'[5], 5, \text{ART})) \\ &= \frac{1}{2} (f(a, 1, \text{ART}) + f(\text{the}, 5, \text{ART})). \end{aligned}$$

Here, the symbol  $f_t(s'[p], p, \text{ART})$  refers to the confidence score of the article classifier at position  $p$ , and  $s'[p]$  is the word at position  $p$  of  $s'$ .

Similarly, the disagreement score  $g(s', \text{ART})$  of the article classifier is

$$\begin{aligned} g(s', \text{ART}) &= \max(g_1, g_2) \\ g_1 &= \arg \max_k f(k, 1, \text{ART}) - f(a, 1, \text{ART}) \\ g_2 &= \arg \max_k f(k, 5, \text{ART}) - f(\text{the}, 5, \text{ART}) \end{aligned}$$

Putting them together, the weight for  $Z_{\text{Noun},2}^{\text{singular}}$  is:

$$\begin{aligned} w_{\text{Noun},2,\text{singular}} &= \nu_{\text{LM}} h(s', \text{LM}) \\ &+ \frac{\lambda_{\text{ART}}}{2} (f(a, 1, \text{ART}) + f(\text{the}, 5, \text{ART})) \\ &+ \lambda_{\text{PREP}} f(\text{on}, 4, \text{PREP}) \\ &+ \frac{\lambda_{\text{NOUN}}}{2} (f(\text{cat}, 2, \text{NOUN}) + f(\text{mat}, 6, \text{NOUN})) \\ &+ \mu_{\text{ART}} g(s', \text{ART}) \\ &+ \mu_{\text{PREP}} g(s', \text{PREP}) \\ &+ \mu_{\text{NOUN}} g(s', \text{NOUN}) \end{aligned}$$

Input	A	cats	sat	on	the	mat
Corrections	The, $\epsilon$	cat		at, in	a, $\epsilon$	mats
	$Z_{\text{Art},1}^a$	$Z_{\text{Noun},2}^{\text{singular}}$		$Z_{\text{Prep},4}^{\text{on}}$	$Z_{\text{Art},5}^a$	$Z_{\text{Noun},6}^{\text{singular}}$
Variables	$Z_{\text{Art},1}^{\text{the}}$	$Z_{\text{Noun},2}^{\text{plural}}$		$Z_{\text{Prep},4}^{\text{at}}$	$Z_{\text{Art},5}^{\text{the}}$	$Z_{\text{Noun},6}^{\text{plural}}$
	$Z_{\text{Art},1}^\epsilon$			$Z_{\text{Prep},4}^{\text{in}}$	$Z_{\text{Art},5}^\epsilon$	

Table 2: The possible corrections on example (6).

### 3.5 Complexity

The time complexity of ILP is determined by the number of variables and constraints. Assume that for each sentence position, at most  $K$  classifiers are applicable<sup>2</sup>. The number of variables is  $O(K|s|C(l^*))$ , where  $l^* = \arg \max_{l \in L} C(l)$ . The number of constraints is  $O(K|s|)$ .

## 4 Constraints for Prior Knowledge

### 4.1 Modification Count Constraints

In practice, we usually have some rough gauge of the quality of an input sentence. If an input sentence is mostly grammatical, the system is expected to make few corrections. This requirement can be easily satisfied by adding modification count constraints.

In this work, we constrain the number of modifications according to error types. For the error type  $l$ , a parameter  $N_l$  controls the number of modifications allowed for type  $l$ . For example, the modification count constraint for article corrections is

$$\sum_{p,k} Z_{\text{Art},p}^k \leq N_{\text{Art}}, \quad \text{where } k \neq s[p]. \quad (7)$$

The condition ensures that the correction  $k$  is different from the original word in the input sentence. Hence, the summation only counts real modifications. There are similar constraints for preposition, noun number, and spelling corrections:

$$\sum_{p,k} Z_{\text{Prep},p}^k \leq N_{\text{Prep}}, \quad \text{where } k \neq s[p], \quad (8)$$

$$\sum_{p,k} Z_{\text{Noun},p}^k \leq N_{\text{Noun}}, \quad \text{where } k \neq s[p], \quad (9)$$

$$\sum_{p,k} Z_{\text{Spell},p}^k \leq N_{\text{Spell}}, \quad \text{where } k \neq s[p]. \quad (10)$$

<sup>2</sup>In most cases,  $K = 1$ . An example of  $K > 1$  is a noun that requires changing the word form (between singular and plural) and inserting an article, for which  $K = 2$ .

### 4.2 Article-Noun Agreement Constraints

An advantage of the ILP formulation is that it is relatively easy to incorporate prior linguistic knowledge. We now take article-noun agreement as an example to illustrate how to encode such prior knowledge using linear constraints.

A noun in plural form cannot have *a* (or *an*) as its article. That two Boolean variables  $Z_1$  and  $Z_2$  are mutually exclusive can be handled using a simple inequality  $Z_1 + Z_2 \leq 1$ . Thus, the following inequality correctly enforces article-noun agreement:

$$Z_{\text{Art},p_1}^a + Z_{\text{Noun},p_2}^{\text{plural}} \leq 1, \quad (11)$$

where the article at  $p_1$  modifies the noun at  $p_2$ .

### 4.3 Dependency Relation Constraints

Another set of constraints involves dependency relations, including subject-verb relation and determiner-noun relation. Specifically, for a noun  $n$  at position  $p$ , we check the word  $w$  related to  $n$  via a child-parent or parent-child relation. If  $w$  belongs to a set of verbs or determiners (*are, were, these, all*) that takes a plural noun, then the noun  $n$  is required to be in plural form by adding the following constraint:

$$Z_{\text{Noun},p}^{\text{plural}} = 1. \quad (12)$$

Similarly, if a noun  $n$  at position  $p$  is required to be in singular form due to subject-verb relation or determiner-noun relation, we add the following constraint:

$$Z_{\text{Noun},p}^{\text{singular}} = 1. \quad (13)$$

## 5 Inference with Second Order Variables

### 5.1 Motivation and Definition

To relax the decomposable assumption in Section 3.2, instead of treating each correction separately, one can combine multiple corrections into a single correction by introducing higher order variables.

Consider the sentence *A cat sat on the mat.* When measuring the gain due to  $Z_{\text{Noun},2}^{\text{plural}} = 1$  (change *cat* to *cats*), the weight  $w_{\text{Noun},2,\text{plural}}$  is likely to be small since *A cats* will get a low language model score, a low article classifier confidence score, and a low noun number classifier confidence score. Similarly, the weight  $w_{\text{Art},1,\epsilon}$  of  $Z_{\text{Art},1}^\epsilon$  (delete article *A*) is also likely to be small because of the missing article. Thus, if one considers the two corrections separately, they are both unlikely to appear in the final corrected output.

However, the correction from *A cat sat on the mat.* to *Cats sat on the mat.* should be a reasonable candidate, especially if the context indicates that there are many cats (more than one) on the mat. Due to treating corrections separately, it is difficult to deal with multiple interacting corrections with only first order variables.

In order to include the correction  $\epsilon$  *Cats*, one can use a new set of variables, *second order variables*. To keep symbols clear, let  $Z = \{Z_u | Z_u = Z_{l,p}^k, \forall l, p, k\}$  be the set of first order variables, and  $w_u = w_{l,p,k}$  be the weight of  $Z_u = Z_{l,p}^k$ . Define a second order variable  $X_{u,v}$ :

$$X_{u,v} = Z_u \wedge Z_v, \quad (14)$$

where  $Z_u$  and  $Z_v$  are first order variables:

$$Z_u \triangleq Z_{l_1,p_1}^{k_1}, \quad Z_v \triangleq Z_{l_2,p_2}^{k_2}. \quad (15)$$

The definition of  $X_{u,v}$  states that a second order variable is set to 1 if and only if its two component first order variables are both set to 1. Thus, it combines two corrections into a single correction. In the above example, a second order variable is introduced:

$$X_{u,v} = Z_{\text{Art},1}^\epsilon \wedge Z_{\text{Noun},2}^{\text{plural}},$$

$$s \xrightarrow{X_{u,v}} s' = \text{Cats sat on the mat.}$$

Similar to first order variables, let  $w_{u,v}$  be the weight of  $X_{u,v}$ . Note that definition (2) only depends on the output sentence  $s'$ , and the weight of the second order variable  $w_{u,v}$  can be defined in the same way:

$$w_{u,v} = \nu_{\text{LM}} h(s', \text{LM}) + \sum_{t \in E} \lambda_t f(s', t) + \sum_{t \in E} \mu_t g(s', t). \quad (16)$$

## 5.2 ILP with Second Order Variables

A set of new constraints is needed to enforce consistency between the first and second order variables. These constraints are the linearization of definition (14) of  $X_{u,v}$ :

$$X_{u,v} = Z_u \wedge Z_v \Leftrightarrow \begin{cases} X_{u,v} \leq Z_u \\ X_{u,v} \leq Z_v \\ X_{u,v} \geq Z_u + Z_v - 1 \end{cases} \quad (17)$$

A new objective function combines the weights from both first and second order variables:

$$\max \sum_{l,p,k} w_{l,p,k} Z_{l,p}^k + \sum_{u,v} w_{u,v} X_{u,v}. \quad (18)$$

In our experiments, due to noisy data, some weights of second order variables are small, even if both of its first order variables have large weights and satisfy all prior knowledge constraints. They will affect ILP proposing good corrections. We find that the performance will be better if we change the weights of second order variables to  $w'_{u,v}$ , where

$$w'_{u,v} \triangleq \max\{w_{u,v}, w_u, w_v\}. \quad (19)$$

Putting them together, (20)-(25) is an ILP formulation using second order variables, where  $X$  is the set of all second order variables which will be explained in the next subsection.

$$\max \sum_{l,p,k} w_{l,p,k} Z_{l,p}^k + \sum_{u,v} w'_{u,v} X_{u,v} \quad (20)$$

$$\text{s.t. } \sum_k Z_{l,p}^k = 1, \quad \forall \text{ applicable } l, p \quad (21)$$

$$X_{u,v} \leq Z_u, \quad (22)$$

$$X_{u,v} \leq Z_v, \quad (23)$$

$$X_{u,v} \geq Z_u + Z_v - 1, \forall X_{u,v} \in X \quad (24)$$

$$X_{u,v}, Z_{l,p}^k \in \{0, 1\} \quad (25)$$

## 5.3 Complexity and Variable Selection

Using the notation in section 3.5, the number of second order variables is  $O(|Z|^2) = O(K^2 |s|^2 C(l^*)^2)$  and the number of constraints is  $O(K^2 |s|^2 C(l^*)^2)$ . More generally, for variables with higher order  $h \geq 2$ , the number of variables (and constraints) is  $O(K^h |s|^h C(l^*)^h)$ .

Note that both the number of variables and the number of constraints increase exponentially with increasing variable order. In practice, a small subset of second order variables is sufficient to

Data set	Sentences	Words	Edits
Dev set	939	22,808	1,264
Test set	722	18,790	1,057

Table 3: Overview of the HOO 2011 data sets. Corrections are called *edits* in the HOO 2011 shared task.

achieve good performance. For example, noun number corrections are only coupled with nearby article corrections, and have no connection with distant or other types of corrections.

In this work, we only introduce second order variables that combine article corrections and noun number corrections. Furthermore, we require that the article and the noun be in the same noun phrase. The set  $X$  of second order variables in Equation (24) is defined as follows:

$$X = \{X_{u,v} = Z_u \wedge Z_v | l_1 = \text{Art}, l_2 = \text{Noun}, s[p_1], s[p_2] \text{ are in the same noun phrase}\},$$

where  $l_1, l_2, p_1, p_2$  are taken from Equation (15).

## 6 Experiments

Our experiments mainly focus on two aspects: how our ILP approach performs compared to other grammatical error correction systems; and how the different constraints and the second order variables affect the ILP performance.

### 6.1 Evaluation Corpus and Metric

We follow the evaluation setup in the HOO 2011 shared task on grammatical error correction (Dale and Kilgarriff, 2011). The development set and test set in the shared task consist of conference and workshop papers taken from the Association for Computational Linguistics (ACL). Table 3 gives an overview of the data sets.

System performance is measured by precision, recall, and  $F$  measure:

$$P = \frac{\# \text{ true edits}}{\# \text{ system edits}}, R = \frac{\# \text{ true edits}}{\# \text{ gold edits}}, F = \frac{2PR}{P+R}. \quad (26)$$

The difficulty lies in how to generate the system edits from the system output. In the HOO 2011 shared task, participants can submit system edits directly or the corrected plain-text system output. In the latter case, the official HOO scorer will extract system edits based on the original (ungram-

matical) input text and the corrected system output text, using GNU Wdiff<sup>3</sup>.

Consider an input sentence *The data is similar with test set.* taken from (Dahlmeier and Ng, 2012a). The gold-standard edits are *with*  $\rightarrow$  *to* and  $\epsilon$   $\rightarrow$  *the*. That is, the grammatically correct sentence should be *The data is similar to the test set.* Suppose the corrected output of a system to be evaluated is exactly this perfectly corrected sentence *The data is similar to the test set.* However, the official HOO scorer using GNU Wdiff will automatically extract only one system edit *with*  $\rightarrow$  *to* for this system output. Since this single system edit does not match any of the two gold-standard edits, the HOO scorer returns an F measure of 0, even though the system output is perfectly correct.

In order to overcome this problem, the *Max-Match* ( $M^2$ ) scorer was proposed in (Dahlmeier and Ng, 2012b). Given a set of gold-standard edits, the original (ungrammatical) input text, and the corrected system output text, the  $M^2$  scorer searches for the system edits that have the largest overlap with the gold-standard edits. For the above example, the system edits automatically determined by the  $M^2$  scorer are identical to the gold-standard edits, resulting in an F measure of 1 as we would expect. We will use the  $M^2$  scorer in this paper to determine the best system edits. Once the system edits are found,  $P$ ,  $R$ , and  $F$  are computed using the standard definition (26).

### 6.2 ILP Configuration

#### 6.2.1 Variables

The first order variables are given in Table 1. If the indefinite article correction  $a$  is chosen, then the final choice between  $a$  and  $an$  is decided by a rule-based post-processing step. For each preposition error variable  $Z_{\text{prep},p}^k$ , the correction  $k$  is restricted to a pre-defined confusion set of prepositions which depends on the observed preposition at position  $p$ . For example, the confusion set of *on* is  $\{at, for, in, of\}$ . The list of prepositions corrected by our system is *about, among, at, by, for, in, into, of, on, over, to, under, with, and within*. Only selected positions in a sentence (determined by rules) undergo punctuation correction. The spelling correction candidates are given by a spell checker. We used GNU Aspell<sup>4</sup> in our work.

<sup>3</sup><http://www.gnu.org/software/wdiff/>

<sup>4</sup><http://aspell.net>

## 6.2.2 Weights

As described in Section 3.2, the weight of each variable is a linear combination of the language model score, three classifier confidence scores, and three classifier disagreement scores. We use the Web 1T 5-gram corpus (Brants and Franz, 2006) to compute the language model score for a sentence. Each of the three classifiers (article, preposition, and noun number) is trained with the multi-class confidence weighted algorithm (Cramer et al., 2009). The training data consists of all non-OCR papers in the ACL Anthology<sup>5</sup>, minus the documents that overlap with the HOO 2011 data set. The features used for the classifiers follow those in (Dahlmeier and Ng, 2012a), which include lexical and part-of-speech n-grams, lexical head words, web-scale n-gram counts, dependency heads and children, etc. Over 5 million training examples are extracted from the ACL Anthology for use as training data for the article and noun number classifiers, and over 1 million training examples for the preposition classifier.

Finally, the language model score, classifier confidence scores, and classifier disagreement scores are normalized to take values in  $[0, 1]$ , based on the HOO 2011 development data. We use the following values for the coefficients:  $\nu_{\text{LM}} = 1$  (language model);  $\lambda_t = 1$  (classifier confidence); and  $\mu_t = -1$  (classifier disagreement).

## 6.2.3 Constraints

In Section 4, three sets of constraints are introduced: modification count (MC), article-noun agreement (ANA), and dependency relation (DR) constraints. The values for the modification count parameters are set as follows:  $N_{\text{Art}} = 3$ ,  $N_{\text{Prep}} = 2$ ,  $N_{\text{Noun}} = 2$ , and  $N_{\text{Spell}} = 1$ .

## 6.3 Experimental Results

We compare our ILP approach with two other systems: the beam search decoder of (Dahlmeier and Ng, 2012a) which achieves the best published performance to date on the HOO 2011 data set, and UI Run1 (Rozovskaya et al., 2011) which achieves the best performance among all participating systems at the HOO 2011 shared task. The results are given in Table 4.

The HOO 2011 shared task provides two sets of gold-standard edits: the original gold-standard edits produced by the annotator, and the official gold-

System	Original			Official		
	P	R	F	P	R	F
UI Run1	40.86	11.21	17.59	54.61	14.57	23.00
Beam search	30.28	19.17	23.48	33.59	20.53	25.48
ILP	20.54	27.93	23.67	21.99	29.04	25.03

Table 4: Comparison of three grammatical error correction systems.

standard edits which incorporated corrections proposed by the HOO 2011 shared task participants. All three systems listed in Table 4 use the  $M^2$  scorer to extract system edits. The results of the beam search decoder and UI Run1 are taken from Table 2 of (Dahlmeier and Ng, 2012a).

Overall, ILP inference outperforms UI Run1 on both the original and official gold-standard edits, and the improvements are statistically significant at the level of significance 0.01. The performance of ILP inference is also competitive with the beam search decoder. The results indicate that a grammatical error correction system benefits from corrections made at a whole sentence level, and that joint correction of multiple error types achieves state-of-the-art performance.

Table 5 provides the comparison of the beam search decoder and ILP inference in detail. The main difference between the two is that, except for spelling errors, ILP inference gives higher recall than the beam search decoder, while its precision is lower. This indicates that ILP inference is more aggressive in proposing corrections.

Next, we evaluate ILP inference in different configurations. We only focus on article and noun number error types. Table 6 shows the performance of ILP in different configurations. From the results, MC and DR constraints improve precision, indicating that the two constraints can help to restrict the number of erroneous corrections. Including second order variables gives the best  $F$  measure, which supports our motivation for introducing higher order variables.

Adding article-noun agreement constraints (ANA) slightly decreases performance. By examining the output, we find that although the overall performance worsens slightly, the agreement requirement is satisfied. For example, for the input *We utilize search engine to . . .*, the output without ANA is *We utilize a search engines to . . .* but with ANA is *We utilize the search engines to . . .*, while the only gold edit inserts *a*.

<sup>5</sup><http://aclweb.org/anthology-new/>

Error type	Original						Official					
	Beam search			ILP			Beam search			ILP		
	P	R	F	P	R	F	P	R	F	P	R	F
Spelling	36.84	0.69	1.35	60.00	0.59	1.17	36.84	0.66	1.30	60.00	0.57	1.12
+ Article	19.84	12.59	15.40	18.54	14.75	16.43	22.45	13.72	17.03	20.37	15.61	17.68
+ Preposition	22.62	14.26	17.49	17.61	18.58	18.09	24.84	15.14	18.81	19.24	19.68	19.46
+ Punctuation	24.27	18.09	20.73	20.52	23.50	21.91	27.13	19.58	22.75	22.49	24.98	23.67
+ Noun number	30.28	19.17	23.48	20.54	27.93	23.67	33.59	20.53	25.48	21.99	29.04	25.03

Table 5: Comparison of the beam search decoder and ILP inference. ILP is equipped with all constraints (MC, ANA, DR) and default parameters. Second order variables related to article and noun number error types are also used in the last row.

Setting	Original			Official		
	P	R	F	P	R	F
Art+Nn, 1 <sup>st</sup> ord.	17.19	19.37	18.22	18.59	20.44	19.47
+ MC	17.87	18.49	18.17	19.23	19.39	19.31
+ ANA	17.78	18.39	18.08	19.04	19.11	19.07
+ DR	17.95	18.58	18.26	19.23	19.30	19.26
+ 2 <sup>nd</sup> ord.	18.75	18.88	18.81	20.04	19.58	19.81

Table 6: The effects of different constraints and second order variables.

## 7 Conclusion

In this paper, we model grammatical error correction as a joint inference problem. The inference problem is solved using integer linear programming. We provide three sets of constraints to incorporate additional linguistic knowledge, and introduce a further extension with second order variables. Experiments on the HOO 2011 shared task show that ILP inference achieves state-of-the-art performance on grammatical error correction.

## Acknowledgments

This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

## References

- Francis Bond and Satoru Ikehara. 1996. When and how to disambiguate? countability in machine translation. In *Proceedings of the International Seminar on Multimodal Interactive Disambiguation*.
- Francis Bond, Kentaro Ogura, and Tsukasa Kawaoka. 1995. Noun phrase reference in Japanese-to-English machine translation. In *Proceedings of the 6th International Conference on Theoretical and Methodological Issues in Machine Translation*.
- Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram corpus version 1.1. Technical report, Google Research.
- Koby Crammer, Mark Dredze, and Alex Kulesza. 2009. Multi-class confidence weighted algorithms. In *Proceedings of EMNLP*.
- Daniel Dahlmeier and Hwee Tou Ng. 2011. Grammatical error correction with alternating structure optimization. In *Proceedings of ACL*.
- Daniel Dahlmeier and Hwee Tou Ng. 2012a. A beam-search decoder for grammatical error correction. In *Proceedings of EMNLP*.
- Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *Proceedings of NAACL*.
- Robert Dale and Adam Kilgarriff. 2011. Helping Our Own: The HOO 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*.
- Robert Dale, Ilya Anisimoff, and George Narroway. 2012. HOO 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Innovative Use of NLP for Building Educational Applications*, pages 54–62.
- Michael Gamon. 2010. Using mostly native data to correct errors in learners’ writing. In *Proceedings of NAACL*.

- Michael Gamon. 2011. High-order sequence modeling for language learner error detection. In *Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications*.
- Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(2).
- Julia Heine. 1998. Definiteness predictions for Japanese noun phrases. In *Proceedings of ACL-COLING*.
- Kevin Knight and Ishwar Chander. 1994. Automated postediting of documents. In *Proceedings of AAAI*.
- Xiaohua Liu, Bo Han, Kuan Li, Stephan Hyeonjun Stiller, and Ming Zhou. 2010. SRL-based verb selection for ESL. In *Proceedings of EMNLP*.
- Andre Martins, Noah Smith, and Eric Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of ACL-IJCNLP*.
- Masaki Murata and Makoto Nagao. 1993. Determination of referential property and number of nouns in Japanese sentences for machine translation into English. In *Proceedings of the 5th International Conference on Theoretical and Methodological Issues in Machine Translation*.
- Y. Albert Park and Roger Levy. 2011. Automated whole sentence grammar correction using a noisy channel model. In *Proceedings of ACL*.
- Vasin Punyakanok, Dan Roth, Wen tau Yih, and Dav Zimak. 2005. Learning and inference over constrained output. In *Proceedings of IJCAI*.
- Sebastian Riedel and James Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proceedings of EMNLP*.
- Sebastian Riedel and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of EMNLP*.
- Alla Rozovskaya and Dan Roth. 2011. Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of ACL*.
- Alla Rozovskaya, Mark Sammons, Joshua Gioja, and Dan Roth. 2011. University of Illinois system in HOO text correction shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*.
- Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proceedings of ACL*.
- Joel R. Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *Proceedings of COLING*.
- Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of ACL*.

# Text-Driven Toponym Resolution using Indirect Supervision

Michael Speriosu     Jason Baldrige

Department of Linguistics

University of Texas at Austin

Austin, TX 78712 USA

{speriosu, jbaldrid}@utexas.edu

## Abstract

Toponym resolvers identify the specific locations referred to by ambiguous place-names in text. Most resolvers are based on heuristics using spatial relationships between multiple toponyms in a document, or metadata such as population. This paper shows that text-driven disambiguation for toponyms is far more effective. We exploit document-level geotags to indirectly generate training instances for text classifiers for toponym resolution, and show that textual cues can be straightforwardly integrated with other commonly used ones. Results are given for both 19th century texts pertaining to the American Civil War and 20th century newswire articles.

## 1 Introduction

It has been estimated that at least half of the world's stored knowledge, both printed and digital, has geographic relevance, and that geographic information pervades many more aspects of humanity than previously thought (Petras, 2004; Skupin and Esperbé, 2011). Thus, there is value in connecting linguistic references to places (e.g. placenames) to formal references to places (coordinates) (Hill, 2006). Allowing for the querying and exploration of knowledge in a geographically informed way requires more powerful tools than a keyword-based search can provide, in part due to the ambiguity of toponyms (placenames).

Toponym resolution is the task of disambiguating toponyms in natural language contexts to geographic locations (Leidner, 2008). It plays an essential role in automated geographic indexing and information retrieval. This is useful for historical research that combines age-old geographic issues like territoriality with modern computational tools (Guldi, 2009), studies of the effect of histor-

ically recorded travel costs on the shaping of empires (Scheidel et al., 2012), and systems that convey the geographic content in news articles (Teitler et al., 2008; Sankaranarayanan et al., 2009) and microblogs (Gelernter and Mushegian, 2011).

Entity disambiguation systems such as those of Kulkarni et al. (2009) and Hoffart et al. (2011) disambiguate references to people and organizations as well as locations, but these systems do not take into account any features or measures unique to geography such as physical distance. Here we demonstrate the utility of incorporating distance measurements in toponym resolution systems.

Most work on toponym resolution relies on heuristics and hand-built rules. Some use simple rules based on information from a gazetteer, such as population or administrative level (city, state, country, etc.), resolving every instance of the same toponym type to the same location regardless of context (Ladra et al., 2008). Others use relationships between multiple toponyms in a context (local or whole document) and look for containment relationships, e.g. *London* and *England* occurring in the same paragraph or as the bigram *London, England* (Li et al., 2003; Amitay et al., 2004; Zong et al., 2005; Clough, 2005; Li, 2007; Volz et al., 2007; Jones et al., 2008; Buscaldi and Rosso, 2008; Grover et al., 2010). Still others first identify unambiguous toponyms and then disambiguate other toponyms based on geopolitical relationships with or distances to the unambiguous ones (Ding et al., 2000). Many favor resolutions of toponyms within a local context or document that cover a smaller geographic area over those that are more dispersed (Rauch et al., 2003; Leidner, 2008; Grover et al., 2010; Loureiro et al., 2011; Zhang et al., 2012). Roberts et al. (2010) use relationships learned between people, organizations, and locations from Wikipedia to aid in toponym resolution when such named entities are present, but do not exploit any other textual context.

Most of these approaches suffer from a major weakness: they rely primarily on spatial relationships and metadata about locations (e.g., population). As such, they often require nearby toponyms (including unambiguous or containing toponyms) to resolve ambiguous ones. This reliance can result in poor coverage when the required information is missing in the context or when a document mentions locations that are neither nearby geographically nor in a geopolitical relationship. There is a clear opportunity that most ignore: use non-toponym textual context. Spatially relevant words like *downtown* that are not explicit toponyms can be strong cues for resolution (Hollenstein and Purves, 2012). Furthermore, the connection between non-spatial words and locations has been successfully exploited in data-driven approaches to document geolocation (Eisenstein et al., 2010, 2011; Wing and Baldrige, 2011; Roller et al., 2012) and other tasks (Hao et al., 2010; Pang et al., 2011; Intagorn and Lerman, 2012; Hecht et al., 2012; Louwerse and Benesh, 2012; Adams and McKenzie, 2013).

In this paper, we learn resolvers that use all words in local or document context. For example, the word *lobster* appearing near the toponym *Portland* indicates the location is Portland in Maine rather than Oregon or Michigan. Essentially, we learn a text classifier per toponym. There are no massive collections of toponyms labeled with locations, so we train models indirectly using geo-tagged Wikipedia articles. Our results show these text classifiers are far more accurate than algorithms based on spatial proximity or metadata. Furthermore, they are straightforward to combine with such algorithms and lead to error reductions for documents that match those algorithms' assumptions.

Our primary focus is toponym resolution, so we evaluate on toponyms identified by human annotators. However, it is important to consider the utility of an end-to-end toponym identification and resolution system, so we also demonstrate that performance is still strong when toponyms are detected with a standard named entity recognizer.

We have implemented all the models discussed in this paper in an open source software package called Fieldspring, which is available on GitHub: <http://github.com/utcompling/fieldspring>. Explicit instructions are provided for preparing data and running code to reproduce our results.

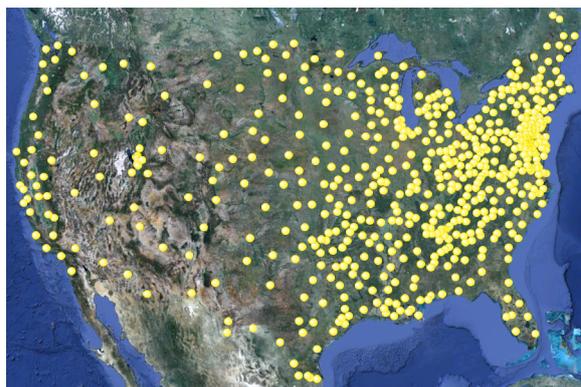


Figure 1: Points representing the United States.

## 2 Data

### 2.1 Gazetteer

Toponym resolvers need a gazetteer to obtain candidate locations for each toponym. Additionally, many gazetteers include other information such as population and geopolitical hierarchy information. We use GEONAMES, a freely available gazetteer containing over eight million entries worldwide.<sup>1</sup> Each location entry contains a name (sometimes more than one) and latitude/longitude coordinates. Entries also include the location's administrative level (e.g. city or state) and its position in the geopolitical hierarchy of countries, states, etc.

GEONAMES gives the locations of regional items like states, provinces, and countries as single points. This is clearly problematic when we seek connections between words and locations: e.g. we might learn that many words associated with the USA are connected to a point in Kansas. To get around this, we represent regional locations as a set of points derived from the gazetteer. Since regional locations are named in the entries for locations they contain, all locations contained in the region are extracted (in some cases over 100,000 of them) and then  $k$ -means is run to find a smaller set of spatial centroids. These act as a tractable proxy for the spatial extent of the entire region.  $k$  is set to the number of  $1^\circ$  by  $1^\circ$  grid cells covered by that region. Figure 1 shows the points computed for the United States.<sup>2</sup> A nice property of this representation is that it does not involve region shape files and the additional programming infrastructure they require.

<sup>1</sup>Downloaded April 16, 2013 from [www.geonames.org](http://www.geonames.org).

<sup>2</sup>The representation also contains three points each in Hawaii and Alaska not shown in Figure 1.

Corpus	docs	toks	types	toks <sub>top</sub>	types <sub>top</sub>	amb <sub>avg</sub>	amb <sub>max</sub>
TRC-DEV	631	136k	17k	4356	613	15.0	857
TRC-DEV-NER	-	-	-	3165	391	18.2	857
TRC-TEST	315	68k	11k	1903	440	13.7	857
TRC-TEST-NER	-	-	-	1346	305	15.7	857
CWAR-DEV	228	33m	200k	157k	850	29.9	231
CWAR-TEST	113	25m	305k	85k	760	31.5	231

Table 1: Statistics of the corpora used for evaluation. Columns subscripted by *top* give figures for toponyms. The last two columns give the average number of candidate locations per toponym token and the number of candidate locations for the most ambiguous toponym.

A location for present purposes is thus a set of points on the earth’s surface. The distance between two locations is computed as the great circle distance between the closest pair of representative points, one from each location.

## 2.2 Toponym Resolution Corpora

We need corpora with toponyms identified and resolved by human annotators for evaluation. The TR-CONLL corpus (Leidner, 2008) contains 946 REUTERS news articles published in August 1996. It has about 204,000 words and articles range in length from a few hundred words to several thousand words. Each toponym in the corpus was identified and resolved by hand.<sup>3</sup> We place every third article into a test portion (TRC-TEST) and the rest in a development portion. Since our methods do not learn from explicitly labeled toponyms, we do not need a training set.

The Perseus Civil War and 19th Century American Collection (CWAR) contains 341 books (58 million words) written primarily about and during the American Civil War (Crane, 2000). Toponyms were annotated by a semi-automated process: a named entity recognizer identified toponyms, and then coordinates were assigned using simple rules and corrected by hand. We divide CWAR into development (CWAR-DEV) and test (CWAR-TEST) sets in the same way as TR-CONLL.

Table 1 gives statistics for both corpora, including the number and ambiguity of gold standard toponyms for both as well as NER identified to-

<sup>3</sup>We found several systematic types of errors in the original TR-CONLL corpus, such as coordinates being swapped for some locations and some longitudes being zero or the negative of their correct values. We repaired many of these errors, though some more idiosyncratic mistakes remain. We, along with Jochen Leidner, will release this updated version shortly and will link to it from our Fieldspring GitHub page.

ponyms for TR-CONLL.<sup>4</sup> We use the pre-trained English NER from the OpenNLP project.<sup>5</sup>

## 2.3 Geolocated Wikipedia Corpus

The GEOWIKI dataset contains over one million English articles from the February 11, 2012 dump of Wikipedia. Each article has human-annotated latitude/longitude coordinates. We divide the corpus into training (80%), development (10%), and test (10%) at random and perform preprocessing to remove markup in the same manner as Wing and Baldrige (2011). The training portion is used here to learn models for text-driven resolvers.

## 3 Toponym Resolvers

Given a set of toponyms provided via annotations or identified using NER, a resolver must select a candidate location for each toponym (or, in some cases, a resolver may abstain). Here, we describe baseline resolvers, a heuristic resolver based on the usual cues used in most toponym resolvers, and several text-driven resolvers. We also discuss combining heuristic and text-driven resolvers.

### 3.1 Baseline Resolvers

**RANDOM** For each toponym, the RANDOM resolver randomly selects a location from those associated in the gazetteer with that toponym.

**POPULATION** The POPULATION resolver selects the location with the greatest population for each toponym. It is generally quite effective, but when a toponym has several locations with large populations, it is often wrong. Also, it can only be used when such information is available, and it is

<sup>4</sup>States and countries are not annotated in CWAR, so we do not evaluate end-to-end using NER plus toponym resolution for it as there are many (falsely) false positives.

<sup>5</sup>[opennlp.apache.org](http://opennlp.apache.org)

less effective if the population statistics are from a time period different from that of the corpus.

### 3.2 SPIDER

Leidner (2008) describes two general and useful *minimality* properties of toponyms:

- *one sense per discourse*: multiple tokens of a toponym in the same text generally do not refer to different locations in the same text
- *spatial minimality*: different toponyms in a text tend refer to spatially near locations

Many toponym resolvers exploit these (Smith and Crane, 2001; Rauch et al., 2003; Leidner, 2008; Grover et al., 2010; Loureiro et al., 2011; Zhang et al., 2012). Here, we define SPIDER (Spatial Prominence via Iterative Distance Evaluation and Reweighting) as a strong representative of such textually unaware approaches. In addition to capturing both minimality properties, it also identifies the relative prominence of the locations for each toponym in a given corpus.

SPIDER resolves each toponym by finding the location for each that minimizes the sum distance to *all* locations for *all* other toponyms in the same document. On the first iteration, it tends to select locations that clump spatially: if *Paris* occurs with *Dallas*, it will choose Paris, Texas even though the topic may be a flight from Texas to France. Further iterations bring Paris, France into focus by capturing its prominence across the corpus. The key intuition is that most documents will discuss Paris, France and only a small portion of these mention places close to Paris, Texas; thus, Paris, France will be selected on the first iteration for many documents (though not for the *Dallas* document). SPIDER thus assigns each candidate location a weight (initialized to 1.0), which is re-estimated on each iteration. The adjusted distance between two locations is computed as the great circle distance divided by the product of the two locations’ weights. At the end of an iteration, each candidate location’s weight is updated to be the fraction of the times it was chosen times the number of candidates for that toponym. The weights are global, with one for each location in the gazetteer, so the same weight vector is used for each token of a given toponym on a given iteration.

For example, if after the first iteration Paris, France is chosen thrice, Paris, Texas once, and Paris, Arkansas never, the global weights of these locations are  $(3/4)*3=2.25$ ,  $(1/4)*3=.75$ , and

$(0/4)*3=0$ , respectively (assume, for the example, there are no other locations named *Paris*). The sum of the weights remains equal to the number of candidate locations. The updated weights are used on the next iteration, so Paris, France will seem “closer” since any distance computed to it is divided by a number greater than one. Paris, Texas will seem somewhat further away, and Paris, Arkansas infinitely far away. The algorithm continues for a fixed number of iterations or until the weights do not change more than some threshold. Here, we run SPIDER for 10 iterations; the weights have generally converged by this point.

When only one toponym is present in a document, we simply select the candidate with the greatest weight. When there is no such weight information, such as when the toponym does not co-occur with other toponyms anywhere in the corpus, we select a candidate at random.

SPIDER captures prominence, but we stress it is not our main innovation: its purpose is to be a benchmark for text-driven resolvers to beat.

### 3.3 Text-Driven Resolvers

The text-driven resolvers presented in this section all use local context windows, document context, or both, to inform disambiguation.

**TRIPDL** We use a document geolocator trained on GEOWIKI’s document location labels. Others—such as Smith and Crane (2001)—have estimated a document-level location to inform toponym resolution, but ours is the first we are aware of to use training data from a different domain to build a document geolocator that uses all words (not only toponyms) to estimate a document’s location. We use the document geolocation method of Wing and Baldrige (2011). It discretizes the earth’s surface into  $1^\circ$  by  $1^\circ$  grid cells and assigns Kullback-Liebler divergences to each cell given a document, based on language models learned for each cell from geolocated Wikipedia articles. We obtain the probability of a cell  $c$  given a document  $d$  by the standard method of exponentiating the negative KL-divergence and normalizing these values over all cells:

$$P(c|d) = \frac{\exp(-KL(c, d))}{\sum_{c'} \exp(-KL(c', d))}$$

This distribution is used for all toponyms  $t$  in  $d$  to define distributions  $P_{DL}(l|t, d)$  over candidate

locations of  $t$  in document  $d$  to be the portion of  $P(c|d)$  consistent with the  $t$ 's candidate locations:

$$P_{DL}(l|t, d) = \frac{P(c_l|d)}{\sum_{l' \in G(t)} P(c_{l'}|d)}$$

where  $G(t)$  is the set of the locations for  $t$  in the gazetteer, and  $c_l$  is the cell containing  $l$ . TRIPDL (Toponym Resolution Informed by Predicted Document Locations) chooses the location that maximizes  $P_{DL}$ .

**WISTR** While TRIPDL uses an off-the-shelf document geolocator to capture the geographic gist of a document, WISTR (Wikipedia Indirectly Supervised Toponym Resolver) instead directly targets each toponym. It learns text classifiers based on local context window features trained on instances automatically extracted from GEOWIKI.

To create the indirectly supervised training data for WISTR, the OpenNLP named entity recognizer detects toponyms in GEOWIKI, and candidate locations for each toponym are retrieved from GEONAMES. Each toponym with a location within 10km of the document location is considered a mention of that location. For example, the *Empire State Building* Wikipedia article has a human-provided location label of (40.75,-73.99). The toponym *New York* is mentioned several times in the article, and GEONAMES lists a *New York* at (40.71,-74.01). These points are 4.8km apart, so each mention of *New York* in the document is considered a reference to New York City.

Next, context windows  $w$  of twenty words to each side of each toponym are extracted as features. The label for a training instance is the candidate location closest to the document location. We extract 1,489,428 such instances for toponyms relevant to our evaluation corpora. These instances are used to train logistic regression classifiers  $P(l|t, w)$  for location  $l$  and toponym  $t$ . To disambiguate a new toponym, WISTR chooses the location that maximizes this probability.

Few such probabilistic toponym resolvers exist in the literature. Li (2007) builds a probability distribution over locations for each toponym, but still relies on nearby toponyms that could refer to regions that contain that toponym and requires hand construction of distributions. Other learning approaches to toponym resolution (e.g. Smith and Mann (2003)) require explicit unambiguous mentions like *Portland, Maine* to construct training instances, while our data gathering methodol-

ogy does not make such an assumption. Overell and Ruger (2008) and Overell (2009) only use nearby toponyms as features. Mani et al. (2010) and Qin et al. (2010) use other word types but only in a local context, and they require toponym-labeled training data. Our approach makes use of all words in local and document context and requires no explicitly labeled toponym tokens.

**TRAWL** We bring TRIPDL, WISTR, and standard toponym resolution cues about administrative levels together with TRAWL (Toponym Resolution via Administrative levels and Wikipedia Locations). The general form of a probabilistic resolver that utilizes such information to select a location  $\hat{l}$  for a toponym  $t$  in document  $d$  may be defined as

$$\hat{l} = \arg \max_l P(l, a_l|t, d).$$

where  $a_l$  is the administrative level (country, state, city) for  $l$  in the gazetteer. This captures the fact that countries (like Sudan) tend to be referred to more often than small cities (like Sudan, Texas). The above term is simplified as follows:

$$\begin{aligned} P(l, a_l|t, d) &= P(a_l|t, d)P(l|a_l, t, d) \\ &\approx P(a_l|t)P(l|t, d) \end{aligned}$$

where we approximate the administrative level prediction as independent of the document, and the location as independent of administrative level. The latter term is then expressed as a linear combination of the local context (WISTR) and the document context (TRIPDL):

$$P(l|t, d) = \lambda_t P(l|t, c_t) + (1-\lambda_t) P_{DL}(l|t, d).$$

$\lambda_t$ , the weight of the local context distribution, is set according to the confidence that a prediction based on local context is correct:

$$\lambda_t = \frac{f(t)}{f(t)+C},$$

where  $f(t)$  is the fraction of training instances of toponym  $t$  of all instances extracted from GEOWIKI.  $C$  is set experimentally;  $C=.0001$  was the optimal value for CWAR-DEV. Intuitively, the larger  $C$  is, the greater  $f(t)$  must be for the local context to be trusted over the document context.

We define  $P(a|t)$ , the administrative level component, to be the fraction of representative points for a location  $\hat{l}$  out of the number of representatives points for all candidate locations  $l \in t$ ,

$$\frac{\|R_{\hat{l}}\|}{\sum_{l' \in t} \|R_{l'}\|}$$

where  $\|R_l\|$  is the number of representative points of  $l$ . This boosts states and countries since higher probability is assigned to locations with more points (and cities have just one point).

Taken together, the above definitions yield the TRAWL resolver, which selects the optimal candidate location  $\hat{l}$  according to

$$\hat{l} = \arg \max_l P(a_i|t)(\lambda_t P(l|t, c_t) + (1-\lambda_t)P_{DL}(l|t, d)).$$

### 3.4 Combining Resolvers and Backoff

SPIDER begins with uniform weights for each candidate location of each toponym. WISTR and TRAWL both output distributions over these locations based on outside knowledge sources, and can be used as more informed initializations of SPIDER than the uniform ones. We call these combinations WISTR+SPIDER and TRAWL+SPIDER.<sup>6</sup>

WISTR fails to predict when encountering a toponym it has not seen in the training data, and TRIPDL fails when a toponym only has locations in cells with no probability mass. TRAWL fails when both of these are true. In these cases, we select the candidate location geographically closest to the most likely cell according to TRIPDL’s  $P(c|d)$  distribution.

### 3.5 Document Size

For SPIDER, runtime is quadratic in the size of documents, so breaking up documents vastly reduces runtime. It also restricts the minimality heuristic—appropriately—to smaller spans of text. For resolvers that take into account the surrounding document when determining how to resolve a toponym, such as TRIPDL and TRAWL, it can often be beneficial to divide documents into smaller subdocuments in order to get a better estimate of the overall geographic prominence of the text surrounding a toponym, but at a more coarse-grained level than the local context models provide. For these reasons, we simply divide each book in the CWAR corpus into small subdocuments of at most 20 sentences.

## 4 Evaluation

Many prior efforts use a simple accuracy metric: the fraction of toponyms whose predicted location

<sup>6</sup>We scale each toponym’s distribution as output by WISTR or TRAWL by the number of candidate locations for that toponym, since the total weight for each toponym in SPIDER is the number of candidate locations, not 1.

is the same as the gold location. Such a metric can be problematic, however. The gazetteer used by a resolver may not contain, for a given toponym, a location whose latitude and longitude *exactly* match the gold label for the toponym (Leidner, 2008). Also, some errors are worse than others, e.g. predicting a toponym’s location to be on the other side of the world versus predicting it to be a different city in the same country—accuracy does not reflect this difference.

We choose a metric that instead measures the distance between the correct and predicted location for each toponym and compute the mean and median of all such error distances. This is used in document geolocation work (Eisenstein et al., 2010, 2011; Wing and Baldrige, 2011; Roller et al., 2012) and is related to the root mean squared distance metric discussed by Leidner (2008).

It is important to understand performance on plain text (without gold toponyms), which is the typical use case for applications using toponym resolvers. Both the accuracy metric and the error-distance metric encounter problems when the set of predicted toponyms is not the same as the set of gold toponyms (regardless of locations), e.g. when a named entity recognizer is used to identify toponyms. In this case, we can use precision and recall, where a true positive is defined as the prediction of a correctly identified toponym’s location to be as close as possible to its gold label, given the gazetteer used. False positives occur when the NER incorrectly predicts a toponym, and false negatives occur when it fails to predict a toponym identified by the annotator. When a correctly identified toponym receives an incorrect location prediction, this counts as both a false negative and a false positive. We primarily present results from experiments with gold toponyms but include an accuracy measure for comparability with results from experiments run on plain text with a named entity recognizer. This accuracy metric simply computes the fraction of toponyms that were resolved as close as possible to their gold label given the gazetteer.

## 5 Results

Table 2 gives the performance of the resolvers on the TR-CONLL and CWAR test sets when gold toponyms are used. Values for RANDOM and SPIDER are averaged over three trials. The ORACLE row gives results when the candidate

Resolver	TRC-TEST			CWAR-TEST		
	Mean	Med.	A	Mean	Med.	A
ORACLE	105	19.8	100.0	0.0	0.0	100.0
RANDOM	3915	1412	33.5	2389	1027	11.8
POPULATION	<b>216</b>	23.1	81.0	1749	<b>0.0</b>	59.7
SPIDER <sub>10</sub>	2180	30.9	55.7	266	<b>0.0</b>	57.5
TRIPDL	1494	29.3	62.0	847	<b>0.0</b>	51.5
WISTR	279	<b>22.6</b>	<b>82.3</b>	855	<b>0.0</b>	69.1
WISTR+SPIDER <sub>10</sub>	430	23.1	81.8	201	<b>0.0</b>	<b>85.9</b>
TRAWL	235	<b>22.6</b>	81.4	945	<b>0.0</b>	67.8
TRAWL+SPIDER <sub>10</sub>	297	23.1	80.7	<b>148</b>	<b>0.0</b>	78.2

Table 2: Accuracy and error distance metrics on test sets with gold toponyms.

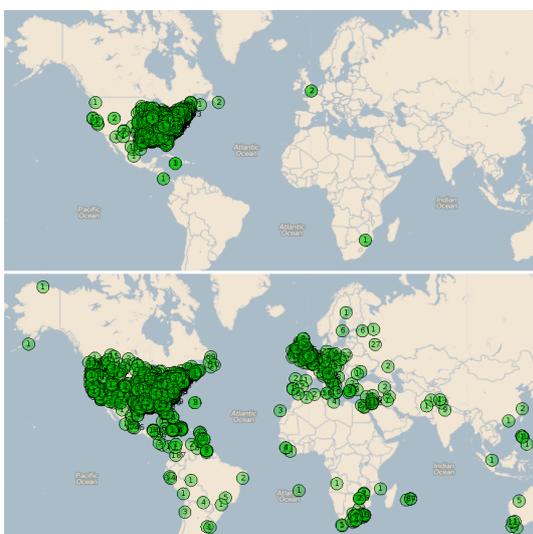


Figure 2: Visualization of how SPIDER clumps most predicted locations in the same region (above), on the CWAR-DEV corpus. TRAWL’s output (below) is much more dispersed.

from GEONAMES closest to the annotated location is always selected. The ORACLE mean and median error values on TR-CONLL are nonzero due to errors in the annotations and inconsistencies stemming from the fact that coordinates from GEONAMES were not used in the annotation of TR-CONLL.

On both datasets, SPIDER achieves errors and accuracies much better than RANDOM, validating the intuition that authors tend to discuss places near each other more often than not, while some locations are more prominent in a given corpus despite violating the minimality heuristic. The text-driven resolvers vastly outperform SPIDER, showing the effectiveness of textual cues for toponym resolution.

The local context resolver WISTR is very effective: it has the highest accuracy for TR-CONLL, though two other text-based resolvers also beat the challenging POPULATION baseline’s accuracy. TRAWL achieves a better mean distance metric for TR-CONLL, and when used to seed SPIDER, it obtains the lowest mean error on CWAR by a large margin. SPIDER seeded with WISTR achieves the highest accuracy on CWAR. The overall geographic scope of CWAR, a collection of documents about the American Civil War, is much smaller than that of TR-CONLL (articles about international events). This makes toponym resolution easier overall (especially error distances) for minimality resolvers like SPIDER, which primarily seek tightly clustered sets of locations. This behavior is quite clear in visualizations of predicted locations such as Figure 2.

On the CWAR dataset, POPULATION performs relatively poorly, demonstrating the fragility of population-based decisions for working with historical corpora. (Also, we note that POPULATION is not a resolver *per se* since it only ever predicts one location for a given toponym, regardless of context.)

Table 3 gives results on TRC-TEST when NER-identified toponyms are used. In this case, the ORACLE results are less than 100% due to the limitations of the NER, and represent the best possible results given the NER we used.

When resolvers are run on NER-identified toponyms, the text-driven resolvers that use local context again easily beat SPIDER. WISTR achieves the best performance. The named entity recognizer is likely better at detecting common toponyms than rare toponyms due to the na-

Resolver	P	R	F
ORACLE	82.6	59.9	69.4
RANDOM	25.1	18.2	21.1
POPULATION	71.6	51.9	60.2
SPIDER <sub>10</sub>	40.5	29.4	34.1
TRIPDL	51.8	37.5	43.5
WISTR	<b>73.9</b>	<b>53.6</b>	<b>62.1</b>
WISTR+SPIDER <sub>10</sub>	73.2	53.1	61.5
TRAWL	72.5	52.5	60.9
TRAWL+SPIDER <sub>10</sub>	72.0	52.2	60.5

Table 3: Precision, recall, and F-score of resolvers on TRC-TEST with NER-identified toponyms.

ture of its training data, and many more local context training instances were extracted from common toponyms than from rare ones in Wikipedia. Thus, our model that uses *only* these local context models does best when running on NER-identified toponyms. We also measured the mean and median error distance for toponyms correctly identified by the named entity recognizer, and found that they tended to be 50-200km worse than for gold toponyms. This also makes sense given the named entity recognizer’s tendency to detect common toponyms: common toponyms tend to be more ambiguous than others.

Results on TR-CONLL indicate much higher performance than the resolvers presented by Leidner (2008), whose F-scores do not exceed 36.5% with either gold or NER toponyms.<sup>7</sup> TRC-TEST is a subset of the documents Leidner uses (he did not split development and test data), but the results still come from overlapping data. The most direct comparison is SPIDER’s F-score of 39.7% compared to his LSW03 algorithm’s 35.6% (both are minimality resolvers). However, our evaluation is more penalized since SPIDER loses precision for NER’s false positives (Jack London as a location) while Leidner only evaluated on actual locations. It thus seems fair to conclude that the text-driven classifiers, with F-scores in the mid-50’s, are much more accurate on the corpus than previous work.

## 6 Error Analysis

Table 4 shows the ten toponyms that caused the greatest total error distances from TRC-DEV with gold toponyms when resolved by TRAWL, the resolver that achieves the lowest mean error on that

<sup>7</sup>Leidner (2008) reports precision, recall, and F-score values even with gold toponyms, since his resolvers can abstain.

dataset among all our resolvers.

*Washington*, the toponym contributing the most total error, is a typical example of a toponym that is difficult to resolve, as there are two very prominent locations within the United States with the name. Choosing one when the other is correct results in an error of over 4000 kilometers. This occurs, for example, when TRAWL chooses Washington state in the phrase *Israel’s ambassador to Washington*, where more knowledge about the status of Washington, D.C. as the political center of the United States (e.g. in the form of more or better contextual training instances) could overturn the administrative level component’s preference for states.

An instance of *California* in a baseball-related news article is incorrectly predicted to be the town California, Pennsylvania. The context is: *...New York starter Jimmy Key left the game in the first inning after Seattle shortstop Alex Rodriguez lined a shot off his left elbow. The Yankees have lost 12 of their last 19 games and their lead in the AL East over Baltimore fell to five games. At California, Tim Wakefield pitched a six-hitter for his third complete game of the season and Mo Vaughn and Troy O’Leary hit solo home runs in the second inning as the surging Boston Red Sox won their third straight 4-1 over the California Angels. Boston has won seven of eight and is 20-6...* The presence of many east coast cues—both toponym and otherwise—make it unsurprising that the resolver would predict California, Pennsylvania despite the administrative level component’s heavier weighting of the state.

The average errors for the toponyms *Australia* and *Russia* are fairly small and stem from differences in how countries are represented across different gazetteers, not true incorrect predictions.

Table 5 shows the toponyms with the greatest errors from CWAR-DEV with gold toponyms when resolved by WISTR+SPIDER. *Rome* is sometimes predicted as cities in Italy and other parts of Europe rather than Rome, Georgia, though it correctly selects the city in Georgia more often than not due to SPIDER’s preference for tightly clumped sets of locations. *Mexico*, however, frequently gets incorrectly selected as a city in Maryland near many other locations in the corpus when TRAWL’s administrative level component is not present. Many other of the toponyms contributing to the total error such as *Jackson* and *Lexington* are

Toponym	N	Mean	Total
Washington	25	3229	80717
Gaza	12	5936	71234
California	8	5475	43797
Montana	3	11635	34905
WA	3	11221	33662
NZ	2	14068	28136
Australia	88	280	24600
Russia	72	260	18712
OR	2	9242	18484
Sydney	12	1422	17067

Table 4: Toponyms with the greatest total error distances in kilometers from TRC-DEV with gold toponyms resolved by TRAWL. N is the number of instances, and the mean error for each toponym type is also given.

Toponym	N	Mean	Total
Mexico	1398	2963	4142102
Jackson	2485	1210	3007541
Monterey	353	2392	844221
Haymarket	41	15663	642170
McMinnville	145	3307	479446
Alexandria	1434	314	450863
Eastport	184	2109	388000
Lexington	796	442	351684
Winton	21	15881	333499
Clinton	170	1401	238241

Table 5: Top errors from CWAR-DEV resolved by TRAWL+SPIDER.

simply the result of many American towns sharing the same names and a lack of clear disambiguating context.

## 7 Conclusion

Our text-driven resolvers prove highly effective for both modern day newswire texts and 19th century texts pertaining to the Civil War. They easily outperform standard minimality toponym resolvers, but can also be combined with them. This strategy works particularly well when predicting toponyms on a corpus with relatively restricted geographic extents. Performance remains good when resolving toponyms identified automatically, indicating that end-to-end systems based on our models may improve the experience of digital humanities scholars interested in finding and visualizing toponyms in large corpora.

## Acknowledgements

We thank: the three anonymous reviewers, Grant DeLozier, and the UT Austin Natural Language Learning reading group, for their helpful feedback; Ben Wing, for his document geolocation software; Jochen Leidner, for providing the TR-CONLL corpus as well as feedback on earlier versions of this paper; and Scott Nesbit, for providing the annotations for the CWAR corpus. This research was supported by a grant from the Morris Memorial Trust Fund of the New York Community Trust.

## References

- B. Adams and G. McKenzie. Inferring thematic places from spatially referenced natural language descriptions. *Crowdsourcing Geographic Knowledge*, pages 201–221, 2013.
- E. Amitay, N. Har’El, R. Sivan, and A. Soffer. Web-a-Where: geotagging web content. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 273–280, 2004.
- D. Buscaldi and P. Rosso. A conceptual density-based approach for the disambiguation of toponyms. *International Journal of Geographical Information Science*, 22(3):301–313, 2008.
- P. Clough. Extracting metadata for spatially-aware information retrieval on the internet. In *Proceedings of the 2005 workshop on Geographic information retrieval*, pages 25–30. ACM, 2005.
- G. Crane. The Perseus Digital Library, 2000. URL <http://www.perseus.tufts.edu>.
- J. Ding, L. Gravano, and N. Shivakumar. Computing geographical scopes of web resources. In *Proceedings of the 26th International Conference on Very Large Data Bases*, pages 545–556, 2000.
- J. Eisenstein, B. O’Connor, N. Smith, and E. Xing. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287, 2010.
- J. Eisenstein, A. Ahmed, and E. Xing. Sparse additive generative models of text. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1041–1048, 2011.

- J. Gelernter and N. Mushegian. Geo-parsing messages from microtext. *Transactions in GIS*, 15(6):753–773, 2011.
- C. Grover, R. Tobin, K. Byrne, M. Woollard, J. Reid, S. Dunn, and J. Ball. Use of the Edinburgh geoparser for georeferencing digitized historical collections. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1925):3875–3889, 2010.
- J. Guldi. The spatial turn. *Spatial Humanities: a Project of the Institute for Enabling*, 2009.
- Q. Hao, R. Cai, C. Wang, R. Xiao, J. Yang, Y. Pang, and L. Zhang. Equip tourists with knowledge mined from travelogues. In *Proceedings of the 19th international conference on World wide web*, pages 401–410, 2010.
- B. Hecht, S. Carton, M. Quaderi, J. Schöning, M. Raubal, D. Gergle, and D. Downey. Explanatory semantic relatedness and explicit spatialization for exploratory search. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 415–424. ACM, 2012.
- L. Hill. *Georeferencing: The Geographic Associations of Information*. MIT Press, 2006.
- J. Hoffart, M. Yosef, I. Bordino, H. Fürstenauf, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics, 2011.
- L. Hollenstein and R. Purves. Exploring place through user-generated content: Using Flickr tags to describe city cores. *Journal of Spatial Information Science*, (1):21–48, 2012.
- S. Intagorn and K. Lerman. A probabilistic approach to mining geospatial knowledge from social annotations. In *Conference on Information and Knowledge Management (CIKM)*, 2012.
- C. Jones, R. Purves, P. Clough, and H. Joho. Modelling vague places with knowledge from the web. *International Journal of Geographical Information Science*, 2008.
- S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of Wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM, 2009.
- S. Ladra, M. Luaces, O. Pedreira, and D. Seco. A toponym resolution service following the OGC WPS standard. In *Web and Wireless Geographical Information Systems*, volume 5373, pages 75–85. 2008.
- J. Leidner. *Toponym resolution in text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names*. Universal Press, Boca Raton, FL, USA, 2008.
- H. Li, R. Srihari, C. Niu, and W. Li. InfoXtract location normalization: a hybrid approach to geographic references in information extraction. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, pages 39–44, 2003.
- Y. Li. Probabilistic toponym resolution and geographic indexing and querying. Master’s thesis, The University of Melbourne, Melbourne, Australia, 2007.
- V. Loureiro, I. Anastácio, and B. Martins. Learning to resolve geographical and temporal references in text. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 349–352, 2011.
- M. Louwerse and N. Benesh. Representing spatial structure through maps and language: Lord of the Rings encodes the spatial structure of Middle Earth. *Cognitive science*, 36(8):1556–1569, 2012.
- I. Mani, C. Doran, D. Harris, J. Hitzeman, R. Quimby, J. Richer, B. Wellner, S. Mardis, and S. Clancy. SpatialML: annotation scheme, resources, and evaluation. *Language Resources and Evaluation*, 44(3):263–280, 2010.
- S. Overell. *Geographic Information Retrieval: Classification, Disambiguation and Modelling*. PhD thesis, Imperial College London, 2009.
- S. Overell and S. Rüger. Using co-occurrence models for placename disambiguation. *International Journal of Geographical Information Science*, 22:265–287, 2008.
- Y. Pang, Q. Hao, Y. Yuan, T. Hu, R. Cai, and L. Zhang. Summarizing tourist destinations by mining user-generated travelogues and pho-

- tos. *Computer Vision and Image Understanding*, 115(3):352 – 363, 2011.
- V. Petras. Statistical analysis of geographic and language clues in the MARC record. Technical report, The University of California at Berkeley, 2004.
- T. Qin, R. Xiao, L. Fang, X. Xie, and L. Zhang. An efficient location extraction algorithm by leveraging web contextual information. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 53–60. ACM, 2010.
- E. Rauch, M. Bukatin, and K. Baker. A confidence-based framework for disambiguating geographic terms. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, pages 50–54, 2003.
- K. Roberts, C. Bejan, and S. Harabagiu. Toponym disambiguation using events. In *Proceedings of the 23rd International Florida Artificial Intelligence Research Society Conference*, pages 271–276, 2010.
- S. Roller, M. Speriosu, S. Rallapalli, B. Wing, and J. Baldrige. Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of EMNLP 2012*, 2012.
- J. Sankaranarayanan, H. Samet, B. Teitler, M. Lieberman, and J. Sperling. TwitterStand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 42–51, 2009.
- W. Scheidel, E. Meeks, and J. Weiland. ORBIS: The Stanford geospatial network model of the roman world. 2012.
- A. Skupin and A. Esperbé. An alternative map of the United States based on an  $n$ -dimensional model of geographic space. *Journal of Visual Languages & Computing*, 22(4):290–304, 2011.
- D. Smith and G. Crane. Disambiguating geographic names in a historical digital library. In *Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries*, pages 127–136, 2001.
- D. Smith and G. Mann. Bootstrapping toponym classifiers. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, pages 45–49, 2003.
- B. Teitler, M. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: a new view on news. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, page 18. ACM, 2008.
- R. Volz, J. Kleb, and W. Mueller. Towards ontology-based disambiguation of geographical identifiers. In *Proceedings of the 16th International Conference on World Wide Web*, 2007.
- B. Wing and J. Baldrige. Simple supervised document geolocation with geodesic grids. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 955–964, 2011.
- Q. Zhang, P. Jin, S. Lin, and L. Yue. Extracting focused locations for web pages. In *Web-Age Information Management*, volume 7142, pages 76–89. 2012.
- W. Zong, D. Wu, A. Sun, E. Lim, and D. Goh. On assigning place names to geography related web pages. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pages 354–362, 2005.

# Argument Inference from Relevant Event Mentions in Chinese Argument Extraction

Peifeng Li, Qiaoming Zhu, Guodong Zhou\*

School of Computer Science & Technology  
Soochow University, Suzhou, 215006, China

{pfli, qmzhu, gdzhou}@suda.edu.cn

## Abstract

As a paratactic language, sentence-level argument extraction in Chinese suffers much from the frequent occurrence of ellipsis with regard to inter-sentence arguments. To resolve such problem, this paper proposes a novel global argument inference model to explore specific relationships, such as *Coreference*, *Sequence* and *Parallel*, among relevant event mentions to recover those inter-sentence arguments in the sentence, discourse and document layers which represent the cohesion of an event or a topic. Evaluation on the ACE 2005 Chinese corpus justifies the effectiveness of our global argument inference model over a state-of-the-art baseline.

## 1 Introduction

The task of event extraction is to recognize event mentions of a predefined event type and their arguments (participants and attributes). Generally, it can be divided into two subtasks: trigger extraction, which aims to identify trigger/event mentions and determine their event type, and argument extraction, which aims to extract various arguments of a specific event and assign the roles to them. In this paper, we focus on argument extraction in Chinese event extraction. While most of previous studies in Chinese event extraction deal with Chinese trigger extraction (e.g., Chen and Ji, 2009a; Qin et al., 2010; Li et al., 2012a, 2012b), there are only a few on Chinese argument extraction (e.g., Tan et al., 2008; Chen and Ji, 2009b). Following previous studies, we divide argument extraction into two components, argument identification

and role determination, where the former recognizes the arguments in a specific event mention and the latter classifies these arguments by roles.

With regard to methodology, most of previous studies on argument extraction recast it as a Semantic Role Labeling (SRL) task and focus on intra-sentence information to identify the arguments and their roles. However, argument extraction is much different from SRL in the sense that, while the relationship between a predicate and its arguments in SRL can be mainly decided from the syntactic structure, the relationship between an event trigger and its arguments are more semantics-based, especially in Chinese, as a paratactic (e.g., discourse-driven and pro-drop) language with the wide spread of ellipsis and the open flexible sentence structure. Therefore, some arguments of a specific event mention are far away from the trigger and how to recover those inter-sentence arguments becomes a challenging issue in Chinese argument extraction. Consider the following discourse (from ACE 2005 Chinese corpus) as a sample:

D1: 巴勒斯坦自治政府否认和加沙走廊 20 号清晨造成两名以色列人**丧生(E1)**的**炸弹攻击(E2)**事件有关...表示将对这起**攻击(E3)**事件展开调查。(The Palestinian National Authority denied any involvement in the **bomb attack (E2)** occurred in the Gaza Strip on the morning of the 20th, which **killed (E1) two Israelites**. ... They claimed that they will be investigating this **attack (E3)**.) - From CBS20001120.1000.0823

In above discourse, there are three event mentions, one *kill* (E1) and two *Attack* (E2, E3). While it is relatively easy to identify 20 号清晨 (morning of 20th), 加沙走廊 (Gaza Strip) and 炸弹 (bomb) as the *Time*, *Place* and *Instrument* roles in E2 by a sentence-based argument

extractor, it is really challenging to recognize these entities as the arguments of its corefered mention E3 since to reduce redundancy in a Chinese discourse, the later Chinese sentences omit many of these entities already mentioned in previous sentences. Similarly, it is hard to recognize 两名以色列人 (two Israelites) as the *Target* role for event mention E2 and identify 炸弹 (bomb) as the *Instrument* role for event mention E1. An alternative way is to employ various relationships among relevant event mentions in a discourse to infer those inter-sentence arguments.

The contributions of this paper are:

- 1) We propose a novel global argument inference model, in which various kinds of event relations are involved to infer more arguments on their semantic relations.
- 2) Different from Liao and Grishman (2010) and Hong et al. (2011), which only consider document-level consistency, we propose a more fine-grained consistency model to enforce the consistency in the sentence, discourse and document layers.
- 3) We incorporate argument semantics into our global argument inference model to unify the semantics of the event and its arguments.

The rest of this paper is organized as follows. Section 2 overviews the related work. Section 3 describes a state-of-the-art Chinese argument extraction system as the baseline. Section 4 introduces our global model in inferring those inter-sentence arguments. Section 5 reports experimental results and gives deep analysis. Finally, we conclude our work in Section 6.

## 2 Related Work

Almost all the existing studies on argument extraction concern English. While some apply pattern-based approaches (e.g., Riloff, 1996; Califf and Mooney, 2003; Patwardhan and Riloff, 2007; Chambers and Jurafsky, 2011), the others use machine learning-based approaches (e.g., Grishman et al., 2005; Ahn, 2006; Patwardhan and Riloff, 2009; Lu and Roth, 2012), most of which rely on various kinds of features in the context of a sentence. In comparison, there are only a few studies exploring inter-sentence information or argument semantics (e.g., Liao and Grishman, 2010; Hong et al., 2011; Huang and Riloff, 2011, 2012).

Compared with the tremendous work on English event extraction, there are only a few studies (e.g., Tan et al., 2008; Chen and Ji, 2009b;

Fu et al., 2010; Qin et al., 2010; Li et al., 2012) on Chinese event extraction with focus on either feature engineering or trigger expansion, under the same framework as English trigger identification. In addition, there are only very few of them focusing on Chinese argument extraction and almost all aim to feature engineering and are based on sentence-level information and recast this task as an SRL-style task. Tan et al. (2008) introduce multiple levels of patterns to improve the coverage in Chinese argument classification. Chen and Ji (2009b) apply various kinds of lexical, syntactic and semantic features to address the special issues in Chinese argument extraction. Fu et al. (2010) use a feature weighting scheme to re-weight various features for Chinese argument extraction. Li et al. (2012b) introduce more refined features to the system of Chen and Ji (2009b) as their baseline.

Specially, several studies have successfully incorporated cross-document or document-level information and argument semantics into event extraction, most of them focused on English.

Yangarber et al. (2007) apply a cross-document inference mechanism to refine local extraction results for the disease name, location and start/end time. Mann (2007) proposes some constraints on relationship rescoring to impose the discourse consistency on the CEO's personal information. Chambers and Jurafsky (2008) propose a narrative event chain which are partially ordered sets of event mentions centered around a common protagonist and this chain can represent the relationship among the relevant event mentions in a document.

Ji and Grishman (2008) employ a rule-based approach to propagate consistent triggers and arguments across topic-related documents. Liao and Grishman (2010) mainly focus on employing the cross-event consistency information to improve sentence-level trigger extraction and they also propose an inference method to infer the arguments following role consistency in a document. Hong et al. (2011) employ the background information to divide an entity type into more cohesive subtypes to create the bridge between two entities and then infer arguments and their roles using cross-entity inference on the subtypes of entities. Huang and Riloff (2012) propose a sequentially structured sentence classifier which uses lexical associations and discourse relations across sentences to identify event-related document contexts and then apply it to recognize arguments and their roles on the relation among triggers and arguments.

### 3 Baseline

In the task of event extraction as defined in ACE evaluations, an event is defined as a specific occurrence involving participants (e.g., *Person, Attacker, Agent, Defendant*) and attributes (e.g., *Place, Time*). Commonly, an event mention is triggered via a word (trigger) in a phrase or sentence which clearly expresses the occurrence of a specific event. The arguments are the entity mentions involved in an event mention with a specific role, the relation of an argument to an event where it participates. Hence, extracting an event consists of four basic steps, identifying an event trigger, determining its event type, identifying involved arguments (participants and attributes) and determining their roles.

As the baseline, we choose a state-of-the-art Chinese event extraction system, as described in Li et al. (2012b), which consists of four typical components: trigger identification, event type determination, argument identification and role determination. In their system, the former two components, trigger identification and event type determination, are processed in a joint model, where the latter two components are run in a pipeline way. Besides, the Maximum-Entropy (ME) model is employed to train individual component classifiers for above four components.

This paper focuses on argument identification and role determination. In order to provide a stronger baseline, we introduce more refined features in such two components, besides those adopted in Li et al. (2012b). Following is a list of features adopted in our baseline.

- 1) Basic features: trigger, POS (Part Of Speech) of the trigger, event type, head word of the entity, entity type, entity subtype;
- 2) Neighbouring features: left neighbouring word of the entity + its POS, right neighbour word of the entity + its POS, left neighbour word of the trigger + its POS, right neighbour word of the trigger + its POS;
- 3) Dependency features: dependency path from the entity to the trigger, depth of the dependency path;
- 4) Syntactic features: path from the trigger to the entity, difference of the depths of the trigger and entity, place of the entity (before trigger or after trigger), depth of the path from the trigger to the entity, siblings of the entity;
- 5) Semantic features: semantic role of the entity tagged by an SRL tool (e.g., *ARG0, ARG1*) (Li et al., 2010), sememe of trigger in Hownet (Dong and Dong, 2006).

### 4 Inferring Inter-Sentence Arguments on Relevant Event Mentions

In this paper, a global argument inference model is proposed to infer those inter-sentence arguments and their roles, incorporating with semantic relations between relevant event mention pairs and argument semantics.

#### 4.1 Motivation

It's well-known that Chinese is a paratactic language, with an open flexible sentence structure and often omits the subject or the object, while English is a hypotactic language with a strict sentence structure and emphasizes on cohesion between clauses. Hence, there are two issues in Chinese argument extraction, associated with its nature of the paratactic language.

The first is that many arguments of an event mention are out of the event mention scope since ellipsis is a common phenomenon in Chinese. We call them inter-sentence arguments in this paper. Table 1 gives the statistics of intra-sentence and inter-sentence arguments in the ACE 2005 Chinese corpus and it shows that 20.8% of the arguments are inter-sentence ones while this figure is less than 1% of the ACE 2005 English corpus. The main reason of that difference is that some Chinese arguments are omitted in the same sentence of the trigger since Chinese is a paratactic language with the wide spread of ellipsis. Besides, a Chinese sentence does not always end with a full stop. In particular, a comma is used frequently as the stop sign of a sentence in Chinese. We detect sentence boundaries, relying on both full stop and comma signs, since in a Chinese document, comma can be also used to sign the end of a sentence. In particular, we detect sentence boundaries on full stop, exclamatory mark and question mark firstly. Then, we identify the sentence boundaries on comma, using a binary classifier with a set of lexical and constituent-based syntactic features, similar to Xue and Yang (2010).

Category	Number
#Arguments	8032
#Inter-sentence	1673(20.8%)
#Intra-sentence	6359(79.2%)

Table 1. Statistics: Chinese argument extraction with regard to intra- sentence and inter-sentence arguments.

The second issue is that the Chinese word order in a sentence is rather agile for the open

flexible sentence structure. Hence, different word orders can often express the same semantics. For example, a *Die* event mention “Three person died in this accident.” can be expressed in many different orders in Chinese, such as “在事故中三人死亡。”, “事故中死亡三人。”, “三人在事故中死亡。”, etc.

In a word, above two issues indicate that syntactic feature-based approaches are limited in identifying Chinese arguments and it will lead to low recall in argument identification. Therefore, employing those high level information to capture the semantic relation, not only the syntactic structure, between the trigger and its long distance arguments is the key to improve the performance of the Chinese argument identification. Unfortunately, it is really hard to find their direct relations since they always appear in different clauses or sentences. An alternative way is to link the different event mentions with their predicates (triggers) and use the trigger as a bridge to connect the arguments to the trigger in another event mention indirectly. Hence, the semantic relations among event mentions are helpful to be a bridge to identify those inter-sentence arguments.

## 4.2 Relations of Event Mention Pairs

In a discourse, most event mentions are surrounding a specific topic. It's obvious that those mentions have the intrinsic relationships to reveal the essential structure of a discourse. Those relevant semantics-based relations are helpful to infer the arguments for a specific trigger mention when the syntactic relations in Chinese argument extraction are not as effective as that in English. In this paper, we divide the relations among relevant event mentions into three categories: *Coreference*, *Sequence* and *Parallel*.

An event may have more than one mention in a document and *coreference* event mentions refer to the same event, as same as the definition in the ACE evaluations. Those *coreference* event mentions always have the same arguments and roles. Therefore, employing this relation can infer the arguments of an event mention from their *Coreference* ones. For example, we can recover the *Time*, *Place* and *Instrument* for E3 via its *Coreference* mention E2 in discourse D1, mentioned in Section 1.

Li et al. (2012a) find out that sometimes two trigger mentions are within a Chinese word whose morphological structure is *Coordination*.

Take the following sentence as a sample:

D2: 一名 17 岁的少年劫持一辆巴士, **刺(E4)死(E5)** 一名妇女。(A 12-year-old younger hijacked a bus and then **stabbed (E4)** a woman to **death (E5)**.) - From ZBN20001218.0400.0005

In D2, 刺死 (stab a person to death) is a trigger with the *Coordination* structure and can be divided into two single-morpheme words 刺 (stab) and 死 (die) while the former triggers an *Attack* event and the latter refers to a *Die* one. It's interesting that they share all arguments in this sentence. The relation between those event mentions whose triggers merge a Chinese word or share the subject and the object are *Parallel*. For the errors in the syntactic parsing, the second single-morpheme trigger is often assigned a wrong tag (e.g., NN, JJ) and this leads to the errors in the argument extraction. Therefore, inferring the arguments of the second single-morpheme trigger from that of the first one based on *Parallel* relation is also an available way to recover arguments.

Like that the topic is an axis in a discourse, the relations among those relevant event mentions with the different types is the bone to link them into a narration. There are a few studies on using the event relations in NLP (e.g., summarization (Li et al., 2006), learning narrative event chains (Chambers and Jurafsky, 2007)) to ensure its effectiveness. In this paper, we define two types of *Sequence* relations of relevant event mentions: *Cause* and *Temporal* for their high probabilities of sharing arguments.

The *Cause* relation between the event mentions are similar to that in the Penn Discourse TreeBank 2.0 (Prasad et al., 2008). For example, an *Attack* event often is the cause of an *Die* or *Injure* event. Our *Temporal* relation is limited to those mentions with the same or relevant event types (e.g., *Transport* and *Arrest*) for the high probabilities of sharing arguments. Take the following discourse as a sample:

D3: 这批战俘离开(E6)阿尔及利亚西部城市廷杜夫前往(E7)摩洛哥西南部城市阿加迪尔。(These prisoners left (E6) Tindouf, a western city of Algeria, and went (E7) to Agadir, a southwestern city of Morocco.) - From Xin20001215.2000.0158

In D3, there are two *Transport* mentions and it is natural to infer 阿加迪尔 (Agadir) as the *Destination* role of E6 and 廷杜夫 (Tindouf) as the *Origin* role of E7 via their *Sequence* relation.

### 4.3 Identifying Relations of Event Mention Pairs

Currently, there are only few studies focusing on such area (e.g., Ahn, 2006; Chamber and Jurafsky, 2007; Huang and Riloff, 2012; Do et al., 2012) and their approaches cannot be introduced to our system directly for the language nature and the different goal. We try to achieve a higher accuracy in this stage so that our argument inference can recover more true arguments.

Inspired by Li and Zhou (2012), we also use the morphological structure to identify the *Parallel* relation. Two parallel event mentions with the adjacent trigger mentions  $w_1$  and  $w_2$  must satisfy follows two conditions:

1)  $Morph(w_1, w_2)$  is *Coordination*

2)  $HM(w_1) \in T_i, HM(w_2) \in T_j \quad i \neq j$

where  $Morph(w_1, w_2)$  is a function to recognize the morphological structure of joint word  $w_1 w_2$ ,  $HM(w_i)$  is to identify the head morpheme<sup>1</sup> in word  $w_i$  and  $T_i$  is the set of the head morphemes with  $i$ th event type. These constraints are enlightened by the fact that only Chinese words with *Coordination* structure can be divided into two new words and each word can trigger an event with the different event type<sup>2</sup>. The implementation of  $Morph(w_1, w_2)$  and  $HM(w)$  are described in Li and Zhou (2012).

The *Coreference* relation is divided into two types: *Noun-based Coreference (NC)* and *Event-based Coreference (EC)* while the former always uses a verbal noun to refer to an event mentioned in current or previous sentence and the latter is that an event is mentioned twice or more actually. For example, the relation between E2 and E3 in D1 is *NC* while the trigger of E3 is only a verbal noun without any direct arguments and it refers to E2.

We adopt a simple rule to recognize those *NC* relations: for each event mention whose trigger is a noun and doesn't act as the subject/object, we regard their relation as *NC* if there is another event mention with the same trigger in current or previous sentence.

Inspired by Ahn (2006), we use the following conditions to infer the *EC* relations between two event mentions with the same event type:

- 1) Their trigger mentions refer to the same trigger;
- 2) They have at least one same or similar

<sup>1</sup> It acts as the governing semantic element in a Chinese word.

<sup>2</sup> If they have the same event type, they will be regarded as a single event mention.

subject/object;

- 3) The score of cosine similarity of two event mentions is more than a threshold<sup>3</sup>.

Finally, for the *Sequence* relation, instead of identifying and classifying the relations clearly and correctly, our goal is to identify whether there are relevant event mentions in a long sentence or two adjacent short sentences who share arguments. Algorithm 1 illustrates a knowledge-based approach to identify the *Sequence* event relation in a discourse for any two trigger mentions  $tri_1$  and  $tri_2$  as follows:

---

#### Algorithm 1

---

```
1: input:  $tri_1$  and  $tri_2$  and their type  $et_1$  and  $et_2$ 
2: output: whether their relation is Sequence
3: begin
4:    $hm_1 \leftarrow HM(tri_1); hm_2 \leftarrow HM(tri_2)$ 
5:    $MP \leftarrow FindAllMP(hm_1, et_1, hm_2, et_2)$ 
6:   for any  $mp_i$  in  $MP$ 
7:     if  $ShareArg(mp_i)$  is true then
8:       return true // Sequence
9:     end if
10:  end for
11:  return false
12: end
```

---

In algorithm 1,  $HM(tri)$  is to identify the head morpheme in trigger  $tri$  and  $FindAllMP(hm_1, et_1, hm_2, et_2)$  is to find all event mention pairs in the training set which satisfy the condition that their head morphemes are  $hm_1$  and  $hm_2$ , and their event types are  $et_1$  and  $et_2$  respectively. Besides,  $ShareArg(mp_i)$  is used to identify whether the event mention pair  $mp_i$  sharing at least one argument. In this algorithm, since the relations on the event types are too coarse, we introduce a more fine-gained *Sequence* relation both on the event types and the head morphemes of the triggers which can divide an event type into many subtypes on the head morpheme. Li and Zhou (2012) have ensured the effectiveness of using head morpheme to infer the triggers and our experiment results also show it is helpful for identifying relevant event mentions which aims to the higher accuracy.

### 4.4 Global Argument Inference Model

Our global argument inference model is composed of two steps: 1) training two sentence-based classifiers: argument identifier (AI) and role determiner (RD) that estimate the score of a candidate acts as an argument and belongs to a

<sup>3</sup> The threshold is tuned to 0.78 on the training set.

specific role following Section 3. 2) Using the scores of two classifiers and the event relations in a sentence, a discourse or a document, we perform global optimization to infer those missing or long distance arguments and their roles.

To incorporate those event relations with our global argument inference model, we regard a document as a tree and divide it into three layers: document, discourse and sentence. A document is composed of a set of the discourses while a discourse contains three sentences. Since almost all arguments (~98%) of a specific event mention in the ACE 2005 Chinese corpus appear in the sentence containing the specific event mention and its two adjacent sentences (previous and next sentences), we only consider these three sentences as a discourse to simplify the process of identifying the scope of a discourse.

We incorporate different event relations into our model on the different layer and the goal of our global argument inference model is to achieve the maximized scores over a document on its three layers and two classifiers: AI and RD. The score of document  $D$  is defined as

$$\hat{D} = \arg \max_{X,Y} (\alpha \sum_{I_i \in D} \sum_{S_{<i,j>} \in I_i} \sum_{T_{<i,j,k>} \in S_{<i,j>}} \sum_{A_Z \in T_{<i,j,k>}} \sum_{(1)} (f_I(E_Z)X_Z + (1-f_I(E_Z))(1-X_Z))$$

$$+ (1-\alpha) \sum_{I_i \in D} \sum_{S_{<i,j>} \in I_i} \sum_{T_{<i,j,k>} \in S_{<i,j>}} \sum_{A_Z \in T_{<i,j,k>}} \sum_{m \in R} \sum_{(2)} (f_D(E_Z, R_m)Y_{<Z,m>} + (1-f_D(E_Z, R_m))(1-Y_{<Z,m>}))$$

$$s.t. \quad X_Z \in \{0,1\} \quad (2)$$

$$Y_{<Z,m>} \in \{0,1\} \quad (3)$$

$$X_Z \geq Y_{<Z,m>} \quad \forall m \in R \quad (4)$$

$$X_Z = \sum_{m \in R} Y_{<Z,m>} \quad (5)$$

where  $I_i$  is the  $i$ th discourses in document  $D$ ;  $S_{<i,j>}$  is the  $j$ th sentences in discourse  $I_i$ ;  $T_{<i,j,k>}$  is the  $k$ th event mentions in sentence  $S_{<i,j>}$ ;  $A_{<i,j,k,l>}$  is the  $l$ th candidate arguments in event mention  $T_{<i,j,k>}$ ;  $Z$  is used to denote  $<i,j,k,l>$ ;  $f_I(E_Z)$  is the score of AI identifying entity mention  $E_Z$  as an argument, where  $E_Z$  is the  $l$ th entity of the  $k$ th event mention of the  $j$ th sentence of the  $i$ th discourse in document  $D$ .  $f_D(E_Z, R_m)$  is the score of RD assigning role  $R_m$  to argument  $E_Z$ . Finally,  $X_Z$  and  $Y_{<Z,m>}$  are the indicators denoting whether entity  $E_Z$  is an argument and whether the role  $R_m$  is assigned to entity  $E_Z$  respectively. Besides, Eq. 4 and Eq. 5 are the inferences to enforce that:

- 1) if an entity belongs to a role, it must be an argument;
- 2) if a entity is an argument of a specific event mention, it must have a role.

**Parallel relation:** Sentence-based optimization is used to incorporate the *Parallel* relation of two event mentions into our model and they share all arguments in a sentence. Since different event type may have different role set, each role in a specific event should be mapped to the corresponding role in its *Parallel* event when they have the different event type. For example, the argument “一名 17 岁的少年” (A 12-year-old younger) in D2 acts as the *Attacker* role in the *Attack* event and the *Agent* role in the *Die* event. We learn those role-pairs from the training set and Table 2 shows part of the role relations learning from the training set.

Event type pair	Role pair
Attack-Die	Attacker-Agent; Target-Victim;...
Injure-Die	Agent-Agent; Victim-Victim;...
Transport-Demonstrate	Artifact-Entity; Destination-Place;...

Table 2. Part of role-pairs for those event mention pairs with *Parallel* relation.

To infer the arguments and their roles on the *Parallel* relation, we enforce the consistency on the role-pair as follows:

$$Y_{<i,j,k,l,m>} = Y_{<i,j,k',l',m'>} \quad (6)$$

$$\forall I_i \in D \wedge S_{<i,j>} \in I_i \wedge T_{<i,j,k>} \in S_{<i,j>} \wedge T_{<i,j,k'>} \in S_{<i,j>} \wedge A_{<i,j,k,l>} \in T_{<i,j,k>} \wedge A_{<i,j,k',l'>} \in T_{<i,j,k'>} \wedge <m,m'> \in RP_{et_h \times et_{h'}} \wedge E_{<i,j,k,l>} = E_{<i,j,k',l'>}$$

where  $RP_{et_h \times et_{h'}}$  is the set of role-pairs between two *Parallel* event mention  $et_h$  and  $et_{h'}$  and  $E_{<i,j,k,l>} = E_{<i,j,k',l'>}$  means they refer to the same entity mention. With the transitivity between the indicators  $X$  and  $Y$ , Eq. 6 also enforces the consistency on  $X_{<i,j,k,l>}$  and  $X_{<i,j,k',l'>}$ .

**Coreference relation:** Since the *NC* and *EC* relation between two event mentions are different in the event expression, we introduce the discourse-based optimization for the former and document-based optimization for the latter.

For two *NC* mentions, we ensure that the succeeding mentions can inherit the arguments from the previous one. To enforce this consistency, we just replace all  $f_I(E_Z)$  and  $f_D(E_Z, R_m)$  of the succeeding event mention with that of the previous one, since the previous one have the more context information.

As for two *EC* event mentions, algorithm 2 shows how to create the constraints for our

global argument inference model to infer arguments and roles.

---

### Algorithm 2

---

1: **input:** two event mentions  $T, T'$  and their arguments set  $A$  and  $A'$   
2: **output:** the constraints set  $C$   
3: **begin**  
4:   **for** each argument  $a$  in  $A$  **do**  
5:      $a' \leftarrow \text{FindSim}(a)$   
6:     **if**  $a' \neq \emptyset$  **then**  
7:        $C \leftarrow C \cup \text{Consistency}(Y_a, Y_{a'})$   
8:     **end if**  
9:   **end for**  
10: **end**

---

In algorithm 2, the function  $\text{FindSim}(a)$  is used to find a similar candidate argument  $a'$  in  $A'$  for  $a$ . If it's found, we enforce the consistency of argument  $a$  and  $a'$  in the role by using  $\text{Consistency}(Y_a, Y_{a'})$  where  $Y_a$  and  $Y_{a'}$  are the indicators in Eq. 1. To evaluate the similarity between two candidates  $a$  and  $a'$ , we regard them as similar ones when they are the same word or in the same entity coreference chain. We use a coreference resolution tool to construct the entity coreference chains, as described in Kong et al (2010).

**Sequence relation:** For any two event mentions in a discourse, we use the event type pair with their head morphemes (e.g., *Attack: 炸 (burst) - Die: 死 (die)*, *Trial-Hearing: 审 (trial) - Sentence: 判 (sentence)*) to search the training set and then obtain the probabilities of sharing the arguments as mentioned in algorithm 1. We denoted  $\text{Pro}_{\langle et, et', HM(tri), HM(tri'), R_m, R_m' \rangle}$  as the probability of the trigger mentions  $tri$  and  $tri'$  (their event types are  $et$  and  $et'$  respectively.) sharing an argument whose roles are  $R_m$  and  $R_m'$  respectively. We propose following discourse-based constraint to enforce the consistency between the roles of two arguments, which are related semantically, temporally, causally or conditionally, based on the probability of sharing an argument and the absolute value of the difference between the scores of RD:

$$\begin{aligned}
Y_{\langle i, j, k, l, m \rangle} &= Y_{\langle i, j', k', l', m' \rangle} \\
\forall i \in D \wedge S_{\langle i, j \rangle}, S_{\langle i, j' \rangle} \in I_i \wedge T_{\langle i, j, k \rangle} \in S_{\langle i, j \rangle} \\
&\wedge T_{\langle i, j', k' \rangle} \in S_{\langle i, j' \rangle} \wedge m, m' \in R \wedge E_{\langle i, j, k, l \rangle} = E_{\langle i, j', k', l' \rangle} \\
&\wedge \text{Pro}_{\langle et, et', HM(tri), HM(tri'), R_m, R_m' \rangle} > \delta \wedge \\
&\left| f_D(E_{\langle i, j, k, l \rangle}, R_m) - f_D(E_{\langle i, j', k', l' \rangle}, R_m') \right| > \lambda
\end{aligned} \quad (7)$$

where  $\delta$  and  $\lambda$  are the thresholds learned from the

development set;  $tri$  and  $tri'$  are triggers of  $k$ th and  $k'$ th event mention whose event types are  $et$  and  $et'$  in  $S_{\langle i, j \rangle}$  and  $S_{\langle i, j' \rangle}$  respectively.

## 4.5 Incorporating Argument Semantics into Global Argument Inference Model

We also introduce the argument semantics, which represent the semantic relations of argument-argument pair, argument-role pair and argument-trigger pair, to reflect the cohesion inside an event. Hong et al. (2011) found out that there is a strong argument and role consistency in the ACE 2005 English corpus. Those consistencies also occur in Chinese and they reveal the relation between the trigger and its arguments, and also explore the relation between the argument and its role. Besides, those entities act as non-argument also have the consistency with high probabilities.

To let the global argument inference model combine those knowledges of argument semantics, we compute the prior probabilities  $P(X_{\langle i, j \rangle} = 1)$  and  $P(Y_{\langle i, j, m \rangle} = 1)$  that entity  $en_j$  occurs in a specific event type  $et_i$  as an argument and its role is  $R_m$  respectively. To overcome the sparsity of the entities, we cluster those entities into more cohesive subtype following Hong et al. (2011). Hence, following the independence assumptions described by Berant et al. (2011), we modify the  $f_I(E_Z)$  and  $f_D(E_Z, R_m)$  in Eq. 1 as follows:

$$f_I(E_Z) = \log \frac{P(X_Z = 1 | F_Z)P(X_Z = 1)}{(1 - P(X_Z = 1 | F_Z))P(X_Z = 0)} \quad (8)$$

$$f_D(E_Z, R_m) = \log \frac{P(Y_{\langle Z, m \rangle} = 1 | F_{\langle Z, m \rangle})P(X_{\langle Z, m \rangle} = 1)}{(1 - P(X_{\langle Z, m \rangle} = 1 | F_{\langle Z, m \rangle}))P(X_{\langle Z, m \rangle} = 0)} \quad (9)$$

where  $P(X_Z = 1 | F_Z)$  and  $P(Y_{\langle Z, m \rangle} = 1 | F_{\langle Z, m \rangle})$  are the probabilities from the AI and AD respectively while  $F_Z$  and  $F_{\langle Z, m \rangle}$  are the feature vectors. Besides,  $P(X_{\langle Z, m \rangle} = 1)$  and  $P(X_Z = 1)$  are the prior probabilities learning from the training set.

## 5 Experimentation

In this section, we first describe the experimental settings and the baseline, and then evaluate our global argument inference model incorporating with relevant event mentions and argument semantics to infer arguments and their roles.

### 5.1 Experimental Settings and Baseline

For fair comparison, we adopt the same experimental settings as the state-of-the-art event extraction system (Li et al. 2012b) and all the

evaluations are experimented on the ACE 2005 Chinese corpus. We randomly select 567 documents as the training set and the remaining 66 documents as the test set. Besides, we reserve 33 documents in the training set as the development set and use the ground truth entities, times and values for our training and testing. As for evaluation, we also follow the standards as defined in Li et al. (2012b). Finally, all the sentences in the corpus are divided into words using a Chinese word segmentation tool (*ICTCLAS*)<sup>1</sup> with all entities annotated in the corpus kept. We use *Berkeley Parser*<sup>2</sup> and *Stanford Parser*<sup>3</sup> to create the constituent and dependency parse trees. Besides, the ME tool (*Maxent*)<sup>4</sup> is employed to train individual component classifiers and *lp\_solver*<sup>5</sup> is used to construct our global argument inference model.

Besides, all the experiments on argument extraction are done on the output of the trigger extraction system as described in Li et al. (2012b). Table 3 shows the performance of the baseline trigger extraction system and Line 1 in Table 4 illustrates the results of argument identification and role determination based on this system.

Trigger identification			Event type determination		
P(%)	R(%)	F1	P(%)	R(%)	F1
74.4	71.9	73.1	71.4	68.9	70.2

Table 3. Performance of the baseline on trigger identification and event type determination.

## 5.2 Inferring Arguments on Relevant Event Mentions and Argument Semantics

We develop a baseline system as mentioned in Section 3 and Line 2 in Table 4 shows that it slightly improves the F1-measure by 0.9% over Li et al. (2012b) due to the incorporation of more refined features. This result indicates the limitation of syntactic-based feature engineering.

Before evaluating our global argument inference model, we should identify the event relations between two mentions in a sentence, a discourse or a document. The experimental results show that the accuracies of identifying *NC*, *EC*, *Parallel* and *Sequence* relation are 80.0%, 72.4%, 88.5% and 87.7% respectively. Those results ensure that our simple methods are

effective. Our statistics on the development set shows almost 65% of the event mentions are involved in those *Coreference*, *Parallel* and *Sequence* relations, which occupy 63%, 50%, 9% respectively<sup>6</sup>. Most of the exceptions are isolated event mentions.

System	Argument identification			Argument role determination		
	P(%)	R(%)	F1	P(%)	R(%)	F1
Li et al.(2012b)	59.1	57.2	58.1	55.8	52.1	53.9
Baseline	60.5	57.6	59.0	55.7	53.0	54.4
BIM	59.3	60.1	59.7	54.4	55.2	54.8
BIM+RE	60.2	65.6	62.8	55.0	60.0	57.4
BIM+RE+AS	<b>62.9</b>	<b>66.1</b>	<b>64.4</b>	<b>57.2</b>	<b>60.2</b>	<b>58.7</b>

Table 4. Performance comparison of argument extraction on argument identification and role determination.

Once the classifier AI and RD are trained, we would like to apply our global argument inference model to infer more inter-sentence arguments and roles. To achieve an optimal solution, we formulate the global inference problem as an Integer Linear Program (ILP), which leads to maximize the objective function. ILP is a mathematical method for constraint-based inference to find the optimal values for a set of variables that maximize an objective function in satisfying a certain number of constraints. In the literature, ILP has been widely used in many NLP applications (e.g., Barzilay and Lapata, 2006; Do et al., 2012; Li et al., 2012b).

For our systems, we firstly evaluate the performance of our basic global argument inference model (BIM) with the Eq. 2–5 which enforce the consistency on AI and RD and then introduce the inference on the relevant event mentions (RE) and argument semantics (AS) to BIM. Table 4 shows their results and we can find out that:

- 1) BIM only slightly improves the performance in F1-measure, as the result of more increase in recall (R) than decrease in precision (P). This suggests that those constraints just enforcing the consistency on AI and RD is not effective enough to infer more arguments.
- 2) Compared to the BIM, our model BIM+RE enhances the performance of argument identification and role determination by 3.1% and 2.6% improvement in F1-measure respectively. This suggests the effectiveness

<sup>1</sup> <http://ictclas.org/>

<sup>2</sup> <http://code.google.com/p/berkeleyparser/>

<sup>3</sup> <http://nlp.stanford.edu/software/lex-parser.shtml>

<sup>4</sup> <http://mallet.cs.umass.edu/>

<sup>5</sup> <http://lpsolve.sourceforge.net/5.5/>

<sup>6</sup> 20% of the mentions belongs to both *Coreference* and *Sequence* relations.

of our global argument inference model on the relevant event mentions to infer inter-sentence arguments. Table 5 shows the contributions of the different event relations while the *Sequence* relation gains the highest improvement of argument identification and role determination in F1-measure respectively.

Constraint	Argument identification			Argument role determination		
	P(%)	R(%)	F1	P(%)	R(%)	F1
BIM	59.3	60.1	59.7	54.4	55.2	54.8
+Parallel	<b>+0.6</b>	+0.7	+0.6	+0.4	+0.6	+0.5
+NC	+0.0	+0.8	+0.4	-0.2	+0.6	+0.2
+EC	<b>+0.6</b>	+1.2	+0.9	<b>+0.5</b>	+1.0	+0.7
+ Sequence	-0.3	<b>+2.8</b>	<b>+1.2</b>	-0.2	<b>+2.6</b>	<b>+1.1</b>

Table 5. Contributions of different event relations on argument identification and role determination. (Incremental)

- 3) Our model BIM+ER+AS gains 1.6% improvement for argument identification, and 1.3% for role determination. The results ensure that argument semantics not only can improve the performance of argument identification, but also is helpful to assign a correct role to an argument in role determination.

Table 3 shows 25.6% of trigger mentions introduced into argument extraction are pseudo ones. If we use the golden trigger extraction, our exploration shows that the precision and recall of argument identification can be up to 78.6% and 88.3% respectively. Table 6 shows the performance comparison of argument extraction on AI and RD given golden trigger extraction. Compared to the Baseline, our system improves the performance of argument identification and role determination by 6.4% and 5.8% improvement in F1-measure respectively, largely due to the dramatic increase in recall of 10.9% and 10.4%.

System	Argument identification			Argument role determination		
	P(%)	R(%)	F1	P(%)	R(%)	F1
Baseline	76.2	77.4	76.8	70.4	72.0	71.2
Model2	<b>78.6</b>	<b>88.3</b>	<b>83.2</b>	<b>72.3</b>	<b>82.4</b>	<b>77.0</b>

Table 6. Performance comparison of argument identification and type determination. (Golden trigger extraction)

### 5.3 Discussion

The initiation of our paper is that syntactic features play an important role in current machine learning-based approaches for English

event extraction, however, their effectiveness is much reduced in Chinese. So the improvement of our model for English event extraction is much less than that of Chinese. However, our model can be an effective complement of the sentence-level English argument extraction systems since the performance of argument extraction is still low in English and using discourse-level information is a way to improve its performance, especially for those event mentions whose arguments spread in complex sentences.

Moreover, our exploration shows that our global argument inference model can mine those arguments within a long distance which are unannotated as arguments of a special event mention in the corpus since the annotators just tagged arguments in a narrow scope or omitted a few arguments. Actually, they are the true ones to our knowledge and are more than 30.6% of those pseudo arguments inferred by our model. This ensures that our global argument inference model and those relations among event mentions is helpful to argument extraction.

## 6 Conclusion

In this paper we propose a global argument inference model to extract those inter-sentence arguments due to the nature of Chinese that it is a discourse-driven pro-drop language with the wide spread of ellipsis and the open flexible sentence structure. In particular, we incorporate various kinds of event relations and the argument semantics into the model in the sentence, discourse and document layers which represent the cohesion of an event or a topic. The experimental results ensure that our global argument inference model outperforms the state-of-the-art system.

In future work, we will focus on introducing more semantic information and cross-document information into the global argument inference model to improve the performance of argument extraction.

## Acknowledgments

The authors would like to thank three anonymous reviewers for their comments on this paper. This research was supported by the National Natural Science Foundation of China under Grant No. 61070123, No. 61272260 and No. 61273320, the National 863 Project of China under Grant No. 2012AA011102. The co-author tagged with “\*” is the corresponding author.

## References

- David Ahn. 2006. *The Stages of Event Extraction*. In Proc. COLING/ACL 2006 Workshop on Annotating and Reasoning about Time and Events. Pages 1-8, Sydney, Australia.
- Regina Barzilay and Miralla Lapata. 2006. *Aggregation via Set Partitioning for Natural Language Generation*. In Proc. NAACL 2006, pages 359-366, New York City, NY.
- Jonathan Berant, Ido Dagan and Jacob Goldberger. 2011. *Global Learning of Typed Entailment Rules*. In Proc. ACL 2011, pages 610-619, Portland, OR.
- Mary Elaine Califf and Raymond J. Mooney. 2003. *Bottom-up Relational Learning of Pattern Matching rules for Information Extraction*. Journal of Machine Learning Research, 4:177-210.
- Nathanael Chambers and Dan Jurafsky. 2008. *Unsupervised Learning of Narrative Event Chains*. In Proc. ACL 2008, pages 789-797, Columbus, OH.
- Nathanael Chambers and Dan Jurafsky. 2011. *Template-based Information Extraction without the Templates*. In Proc. ACL 2011, pages 976-986, Portland, OR.
- Zheng Chen and Heng Ji. 2009a. *Can One Language Bootstrap the Other: A Case Study on Event Extraction*. In Proc. NAACL/HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing, pages 66-74, Boulder, Colorado.
- Zheng Chen and Heng Ji. 2009b. *Language Specific Issue and Feature Exploration in Chinese Event Extraction*. In Proc. NAACL HLT 2009, pages 209-212, Boulder, Colorado.
- Zhengdong Dong and Qiang Dong. 2006. *HowNet and the Computation of Meaning*. World Scientific Pub Co. Inc.
- Quang Xuan Do, Wei Lu and Dan Roth. 2012. *Joint Inference for Event Timeline Construction*. In Proc. EMNLP 2012, pages 677-687, Jeju, Korea.
- Jianfeng Fu, Zongtian Liu, Zhaoman Zhong and Jianfang Shan. 2010. *Chinese Event Extraction Based on Feature Weighting*. Information Technology Journal, 9: 184-187.
- Ralph Grishman, David Westbrook and Adam Meyers. 2005. *NYU's English ACE 2005 System Description*. In Proc. ACE 2005 Evaluation Workshop, Gaithersburg, MD.
- Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou and Qiaoming Zhu. 2011. *Using Cross-Entity Inference to Improve Event Extraction*. In Proc. ACL 2011, pages 1127-1136, Portland, OR.
- Ruihong Huang and Ellen Riloff. 2011. *Peeling Back the Layers: Detecting Event Role Fillers in Secondary Contexts*, In Proc. ACL 2011, pages 1137-1147, Portland, OR.
- Ruihong Huang and Ellen Riloff. 2012. *Modeling Textual Cohesion for Event Extraction*. In Proc. AACL 2012, pages 1664-1770, Toronto, Canada.
- Heng Ji and Ralph Grishman. 2008. *Refining Event Extraction through Cross-Document Inference*. In Proc. ACL 2008, pages 254-262, Columbus, OH.
- Fang Kong, Guodong Zhou, Longhua Qian and Qiaoming Zhu. 2010. *Dependency-driven Anaphoricity Determination for Coreference Resolution*. In Proc. COLING 2010, pages 599-607, Beijing, China.
- Junhui Li, Guodong Zhou and Hwee Tou Ng. 2010. *Joint Syntactic and Semantic Parsing of Chinese*. In Proc. ACL 2010, pages 1108-1117, Uppsala, Sweden.
- Peifeng Li, Guodong Zhou, Qiaoming Zhu and Libin Hou. 2012a. *Employing Compositional Semantics and Discourse Consistency in Chinese Event Extraction*. In Proc. EMNLP 2012, pages 1006-1016, Jeju, Korea.
- Peifeng Li, Qiaoming Zhu, Hongjun Diao and Guodong Zhou. 2012b. *Joint Modeling of Trigger Identification and Event Type Determination in Chinese Event Extraction*. In Proc. COLING 2012, pages 1635-1652, Mumbai, India.
- Peifeng Li and Guodong Zhou. 2012. *Employing Morphological Structures and Sememes for Chinese Event Extraction*. In Proc. COLING 2012, pages 1619-1634, Mumbai, India.
- Wenjie Li, Mingliu Wu, Qin Lu, Wei Xu and Chunfa Yuan. 2006. *Extractive Summarization using Inter- and Intra- Event Relevance*. In Proc. COLING/ACL 2006, pages 369-376, Sydney, Australia.
- Shasha Liao and Ralph Grishman. 2010. *Using Document Level Cross-Event Inference to Improve Event Extraction*. In Proc. ACL 2010, pages 789-797, Uppsala, Sweden.
- Wei Lu and Dan Roth. 2012. *Automatic Event Extraction with Structured Preference Modeling*. In Proc. ACL 2012, pages 835-844, Jeju, Korea.
- Gideon Mann. 2007. *Multi-document Relationship Fusion via Constraints on Probabilistic Databases*. In Proc. HLT/NAACL 2007, pages 332-229, Rochester, NY.
- Siddharth Patwardhan and Ellen Riloff. 2007. *Effective Information Extraction with Semantic Affinity Patterns and Relevant Regions*. In Proc. EMNLP/CoNLL 2007, pages 717-727, Prague, Czech Republic.
- Siddharth Patwardhan and Ellen Riloff. 2009. *A Unified Model of Phrasal and Sentential Evidence*

- for Information Extraction*. In Proc. EMNLP 2009, pages 151-160, Singapore.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi and Bonnie Webber. 2008. *The Penn Discourse Treebank 2.0*. In Proc. LREC 2008, pages 2961-2968, Marrakech, Morocco.
- Bing Qin, Yanyan Zhao, Xiao Ding, Ting Liu and Guofu Zhai. 2010. *Event Type Recognition Based on Trigger Expansion*. Tsinghua Science and Technology, 15(3): 251-258, Beijing, China.
- Ellen Riloff. 1996. *Automatically Generating Extraction Patterns from Untagged Text*. In Proc. AAAI 1996, pages 1044-1049, Portland, OR.
- Hongye Tan, Tiejun Zhao, Jiaheng Zheng. 2008. *Identification of Chinese Event and Their Argument Roles*. In Proc. 2008 IEEE International Conference on Computer and Information Technology Workshops, pages 14-19, Sydney, Australia.
- Nianwen Xue and Yaqin Yang. 2010. *Chinese Sentence Segmentation as Comma Classification*. In Proc. ACL 2010, pages 631-635, Uppsala, Sweden.
- Roman Yangarber, Clive Best, Peter von Etter, Flavio Fuart, David Horby and Ralf Steinberger. 2007. *Combining Information about Epidemic Threats from Multiple Sources*. In Proc. RANLP 2007 Workshop on Multi-source, Multilingual Information Extraction and Summarization, pages 41-48, Borovets, Bulgaria.

# Fine-grained Semantic Typing of Emerging Entities

Ndapandula Nakashole, Tomasz Tylenda, Gerhard Weikum

Max Planck Institute for Informatics  
Saarbrücken, Germany

{nnakasho,tylenda,weikum}@mpi-inf.mpg.de

## Abstract

Methods for information extraction (IE) and knowledge base (KB) construction have been intensively studied. However, a largely under-explored case is tapping into highly dynamic sources like news streams and social media, where new entities are continuously emerging. In this paper, we present a method for discovering and semantically typing newly emerging out-of-KB entities, thus improving the freshness and recall of ontology-based IE and improving the precision and semantic rigor of open IE. Our method is based on a probabilistic model that feeds weights into integer linear programs that leverage type signatures of relational phrases and type correlation or disjointness constraints. Our experimental evaluation, based on crowd-sourced user studies, show our method performing significantly better than prior work.

## 1 Introduction

A large number of knowledge base (KB) construction projects have recently emerged. Prominent examples include Freebase (Bollacker 2008) which powers the Google Knowledge Graph, ConceptNet (Havasi 2007), YAGO (Suchanek 2007), and others. These KBs contain many millions of entities, organized in hundreds to hundred thousands of semantic classes, and hundred millions of relational facts between entities. However, despite these impressive advances, there are still major limitations regarding coverage and freshness. Most KB projects focus on entities that appear in Wikipedia (or other reference collections such as IMDB), and very few have tried to gather entities “in the long tail” beyond prominent sources. Vir-

tuallly all projects miss out on newly emerging entities that appear only in the latest news or social media. For example, the Greenlandic singer Nive Nielsen has gained attention only recently and is not included in any KB (a former Wikipedia article was removed because it “does not indicate the importance or significance of the subject”), and the resignation of BBC director Entwistle is a recently new entity (of type event).

**Goal.** Our goal in this paper is to discover emerging entities of this kind on the fly as they become noteworthy in news and social-media streams. A similar theme is pursued in research on *open information extraction (open IE)* (Banko 2007; Fader 2011; Talukdar 2010; Venetis 2011; Wu 2012), which yields higher recall compared to ontology-style KB construction with canonicalized and semantically typed entities organized in prespecified classes. However, state-of-the-art open IE methods extract all noun phrases that are likely to denote entities. These phrases are not canonicalized, so the same entity may appear under many different names, e.g., “Mr. Entwistle”, “George Entwistle”, “the BBC director”, “BBC head Entwistle”, and so on. This is a problem because names and titles are ambiguous, and this hampers precise search and concise results.

Our aim is for all recognized and newly discovered entities to be semantically interpretable by having *fine-grained types* that connect them to KB classes. The expectation is that this will boost the disambiguation of known entity names and the grouping of new entities, and will also strengthen the extraction of relational facts about entities. For informative knowledge, new entities must be typed in a fine-grained manner (e.g., guitar player, blues band, concert, as opposed to crude types like person, organization, event).

Strictly speaking, the new entities that we cap-

ture are *typed noun phrases*. We do not attempt any cross-document co-reference resolution, as this would hardly work with the long-tail nature and sparse observations of emerging entities. Therefore, our setting resembles the established task of fine-grained typing for noun phrases (Fleischmann 2002), with the difference being that we disregard common nouns and phrases for prominent in-KB entities and instead exclusively focus on the difficult case of phrases that likely denote new entities. The baselines to which we compare our method are state-of-the-art methods for noun-phrase typing (Lin 2012; Yosef 2012).

**Contribution.** The solution presented in this paper, called PEARL, leverages a repository of relational patterns that are organized in a type-signature taxonomy. More specifically, we harness the PATTY collection consisting of more than 300,000 typed paraphrases (Nakashole 2012). An example of PATTY’s expressive phrases is:  $\langle \text{musician} \rangle * \text{cover} * \langle \text{song} \rangle$  for a musician performing someone else’s song. When extracting noun phrases, PEARL also collects the co-occurring PATTY phrases. The type signatures of the relational phrases are cues for the type of the entity denoted by the noun phrase. For example, an entity named Snoop Dogg that frequently co-occurs with the  $\langle \text{singer} \rangle * \text{distinctive voice in} * \langle \text{song} \rangle$  pattern is likely to be a singer. Moreover, if one entity in a relational triple is in the KB and can be properly disambiguated (e.g., a singer), we can use a partially bound pattern to infer the type of the other entity (e.g., a song) with higher confidence.

In this line of reasoning, we also leverage the common situation that many input sentences contain one entity registered in the KB and one novel or unknown entity. Known entities are recognized and mapped to the KB using a recent tool for named entity disambiguation (Hoffart 2011). For cleaning out false hypotheses among the type candidates for a new entity, we devised probabilistic models and an integer linear program that considers incompatibilities and correlations among entity types.

In summary, our contribution in this paper is a model for discovering and ontologically typing out-of-KB entities, using a fine-grained type system and harnessing relational paraphrases with type signatures for probabilistic weight computa-

tion. Crowdsourced quality assessments demonstrate the accuracy of our model.

## 2 Detection of New Entities

To detect noun phrases that potentially refer to entities, we apply a part-of-speech tagger to the input text. For a given noun phrase, there are four possibilities: *a)* The noun phrase refers to a general concept (a class or abstract concept), not an individual entity. *b)* The noun phrase is a known entity that can be directly mapped to the knowledge base. *c)* The noun phrase is a new name for a known entity. *d)* The noun phrase is a new entity not known to the knowledge base at all. In this paper, our focus is on case d); all other cases are out of the scope of this paper.

We use an extensive dictionary of surface forms for in-KB entities (Hoffart 2012), to determine if a name or phrase refers to a known entity. If a phrase does not have any match in the dictionary, we assume that it refers to a new entity. To decide if a noun phrase is a true entity (i.e., an individual entity that is a member of one or more lexical classes) or a non-entity (i.e., a common noun phrase that denotes a class or a general concept), we base the decision on the following hypothesis (inspired by and generalizing (Bunescu 2006): A given noun phrase, not known to the knowledge base, is a true entity if its headword is singular and is consistently capitalized (i.e., always spelled with the first letter in upper case).

## 3 Typing Emerging Entities

To deduce types for new entities we propose to align new entities along the type signatures of patterns they occur with. In this manner we use the patterns to suggest types for the entities they occur with. In particular, we infer entity types from pattern type signatures. Our approach builds on the following hypothesis:

### Hypothesis 3.1 (Type Alignment Hypothesis)

*For a given pattern such as  $\langle \text{actor} \rangle$ ’s character in  $\langle \text{movie} \rangle$ , we assume that an entity pair  $(x, y)$  frequently occurring with the pattern in text implies that  $x$  and  $y$  are of the types  $\langle \text{actor} \rangle$  and  $\langle \text{movie} \rangle$ , respectively.*

**Challenges and Objective.** While the type alignment hypothesis works as a starting point, it introduces false positives. Such false positives stem

from the challenges of polysemy, fuzzy pattern matches, and incorrect paths between entities. With polysemy, the same lexico-syntactic pattern can have different type signatures. For example, the following are three different patterns:  $\langle \text{singer} \rangle \text{released} \langle \text{album} \rangle$ ,  $\langle \text{music\_band} \rangle \text{released} \langle \text{album} \rangle$ ,  $\langle \text{company} \rangle \text{released} \langle \text{product} \rangle$ . For an entity pair  $(x, y)$  occurring with the pattern “released”,  $x$  can be one of three different types.

We cannot expect that the phrases we extract in text will be exact matches of the typed relational patterns learned by PATTY. Therefore, for better recall, we must accept fuzzy matches. Quite often however, the extracted phrase matches multiple relational patterns to various degrees. Each of the matched relational patterns has its own type signature. The type signatures of the various matched patterns can be incompatible with one another.

The problem of incorrect paths between entities emerges when a pair of entities occurring in the same sentence do not stand in a true subject-object relation. Dependency parsing does not adequately solve the issue. Web sources contain a plethora of sentences that are not well-formed. Such sentences mislead the dependency parser to extract wrong dependencies.

Our solution takes into account polysemy, fuzzy matches, as well as issues stemming from potential incorrect-path limitations. We define and solve the following optimization problem:

**Definition 1 (Type Inference Optimization)**

*Given all the candidate types for  $x$ , find the best types or “strongly supported” types for  $x$ . The final solution must satisfy type disjointness constraints. Type disjointness constraints are constraints that indicate that, semantically, a pair of types cannot apply to the same entity at the same time. For example, a  $\langle \text{university} \rangle$  cannot be a  $\langle \text{person} \rangle$ .*

We also study a relaxation of type disjointness constraints through the use of type correlation constraints. Our task is therefore twofold: first, generate candidate types for new entities; second, find the best types for each new entity among its candidate types.

**4 Candidate Types for Entities**

For a given entity, candidate types are types that can potentially be assigned to that entity, based on

the entity’s co-occurrences with typed relational patterns.

**Definition 2 (Candidate Type)** *Given a new entity  $x$  which occurs with a number of patterns  $p_1, p_2, \dots, p_n$ , where each pattern  $p_i$  has a type signature with a domain and a range: if  $x$  occurs on the left of  $p_i$ , we pick the domain of  $p_i$  as a candidate type for  $x$ ; if  $x$  occurs on the right of  $p_i$ , we pick the range of  $p_i$  as a candidate type for  $x$ .*

For each candidate type, we compute confidence weights. Ideally, if an entity occurs with a pattern which is highly specific to a given type then the candidate type should have high confidence. For example “is married to” is more specific to people than “expelled from”. A *person* can be expelled from an *organization* but a *country* can also be expelled from an *organization* such as NATO. There are various ways to compute weights for *candidate types*. We first introduce a uniform weight approach and then present a method for computing more informative weights.

**4.1 Uniform Weights**

We are given a new entity  $x$  which occurs with phrases  $(x \text{ phrase}_1 y_1), (x \text{ phrase}_2 y_2), \dots, (x \text{ phrase}_n y_n)$ . Suppose these occurrences lead to the facts  $(x, p_1, y_1), (x, p_2, y_2), \dots, (x, p_n, y_n)$ . The  $p_i$ s are the *typed relational patterns* extracted by PATTY. The facts are generated by matching *phrases* to relational patterns with type signatures. The type signature of a pattern is denoted by:

$$\text{sig}(p_i) = (\text{domain}(p_i), \text{range}(p_i))$$

We allow fuzzy matches, hence each fact comes with a match score. This is the similarity degree between the phrase observed in text and the typed relational pattern.

**Definition 3 (Fuzzy Match Score)** *Suppose we observe the surface string:  $(x \text{ phrase } y)$  which leads to the fact:  $x, p_i, y$ . The fuzzy match similarity score is:  $\text{sim}(\text{phrase}, p_i)$ , where similarity is the  $n$ -gram Jaccard similarity between the phrase and the typed pattern.*

The confidence that  $x$  is of type *domain* is defined as follows:

**Definition 4 (Candidate Type Confidence)**

*For a given observation  $(x \text{ phrase } y)$ , where*

phrase matches patterns  $p_1, \dots, p_n$ , with domains  $d_1, \dots, d_b$  which are possibly the same:

$$\text{typeConf}(x, \text{phrase}, d) = \sum_{\{p_i: \text{domain}(p_i)=d\}} (\text{sim}(\text{phrase}, p_i))$$

Observe that this sums up over all patterns that match the phrase.

To compute the final confidence for  $\text{typeConf}(x, \text{domain})$ , we aggregate the confidences over all phrases occurring with  $x$ .

**Definition 5 (Aggregate Confidence)** For a set of observations  $(x, \text{phrase}_1, y_1), (x, \text{phrase}_2, y_2), \dots, (x, \text{phrase}_n, y_n)$ , the aggregate candidate type confidence is given by:

$$\begin{aligned} \text{aggTypeConf}(x, d) &= \sum_{\text{phrase}_i} \text{typeConf}(x, \text{phrase}_i, d) \\ &= \sum_{\text{phrase}_i} \sum_{\{p_j: \text{domain}(p_j)=d\}} (\text{sim}(\text{phrase}_i, p_j)) \end{aligned}$$

The confidence for the range  $\text{typeConf}(x, \text{range})$  is computed analogously. All confidence weights are normalized to values in  $[0, 1]$ .

The limitation of the uniform weight approach is that each pattern is considered equally good for suggesting candidate types. Thus this approach does not take into account the intuition that an entity occurring with a pattern which is highly specific to a given type is a stronger signal that the entity is of the type suggested. Our next approach addresses this limitation.

## 4.2 Co-occurrence Likelihood Weight Computation

We devise a likelihood model for computing weights for entity candidate types. Central to this model is the estimation of the likelihood of a given type occurring with a given pattern.

Suppose using PATTY methods we mined a typed relational pattern  $\langle t_1 \rangle p \langle t_2 \rangle$ . Suppose that we now encounter a new entity pair  $(x, y)$  occurring with a phrase that matches  $p$ . We can compute the likelihood of  $x$  and  $y$  being of types  $t_1$  and  $t_2$ , respectively, from the likelihood of  $p$  co-occurring with entities of types  $t_1, t_2$ . Therefore we are interested in the type-pattern likelihood, defined as follows:

**Definition 6 (Type-Pattern Likelihood)** The likelihood of  $p$  co-occurring with an entity pair  $(x, y)$  of the types  $(t_1, t_2)$  is given by:

$$P[t_1, t_2|p] \quad (1)$$

where  $t_1$  and  $t_2$  are the types of the arguments observed with  $p$  from a corpus such as Wikipedia.  $P[t_1, t_2|p]$  is expanded as follows:

$$P[t_1, t_2|p] = \frac{P[t_1, t_2, p]}{P[p]} \quad (2)$$

The expressions on the right-hand side of Equation 2 can be directly estimated from a corpus. We use Wikipedia (English), for corpus-based estimations.  $P[t_1, t_2, p]$  is the relative occurrence frequency of the typed pattern among all entity-pattern-entity triples in a corpus (e.g., the fraction of  $\langle \text{musician} \rangle$  plays  $\langle \text{song} \rangle$  among all triples).  $P[p]$  is the relative occurrence frequency of the untyped pattern (e.g., plays) regardless of the argument types. For example, this sums up over both  $\langle \text{musician} \rangle$  plays  $\langle \text{song} \rangle$  occurrences and  $\langle \text{actor} \rangle$  plays  $\langle \text{fictional character} \rangle$ . If we observe a fact where one argument name can be easily disambiguated to a knowledge-base entity so that its type is known, and the other argument is considered to be an out-of-knowledge-base entity, we condition the joint probability of  $t_1, p$ , and  $t_2$  in a different way:

**Definition 7 (Conditional Type-Pattern Likelihood)**

The likelihood of an entity of type  $t_1$  occurring with a pattern  $p$  and an entity of type  $t_2$  is given by:

$$P[t_1|t_2, p] = \frac{P[t_1, t_2, p]}{P[p, t_2]} \quad (3)$$

where the  $P[p, t_2]$  is the relative occurrence frequency of a partial triple, for example,  $\langle * \rangle$  plays  $\langle \text{song} \rangle$ .

Observe that all numbers refer to occurrence frequencies. For example,  $P[t_1, p, t_2]$  is a fraction of the total number of triples in a corpus.

Multiple patterns can suggest the same type for an entity. Therefore, the weight of the assertion that  $y$  is of type  $t$ , is the total support strength from all phrases that suggest type  $t$  for  $y$ .

**Definition 8 (Aggregate Likelihood)** The aggregate likelihood candidate type confidence is given

by:

$$\text{typeConf}(x, \text{domain}) = \sum_{\text{phrase}_i} \sum_{p_j} \left( \text{sim}(\text{phrase}_i, p_j) * \Upsilon \right)$$

Where  $\Upsilon = P[t_1, t_2|p]$  or  $P[t_1|t_2, p]$  or  $P[t_2|t_1, p]$

The confidence weights are normalized to values in  $[0, 1]$ . So far we have presented a way of generating a number of *weighted candidate types* for  $x$ . In the next step we pick the best types for an entity among all its candidate types.

### 4.3 Integer Linear Program Formulation

Given a set of *weighted candidate types*, our goal is to pick a compatible subset of types for  $x$ . The additional asset that we leverage here is the compatibility of types: how likely is it that an entity belongs to both type  $t_i$  and type  $t_j$ . Some types are mutually exclusive, for example, the type *location* rules out *person* and, at finer levels, *city* rules out *river* and *building*, and so on. Our approach harnesses these kinds of constraints. Our solution is formalized as an Integer Linear Program (ILP). We have candidate types for  $x$ :  $t_1, \dots, t_n$ . First, we define a decision variable  $T_i$  for each candidate type  $i = 1, \dots, n$ . These are binary variables:  $T_i = 1$  means type  $t_i$  is selected to be included in the set of types for  $x$ ,  $T_i = 0$  means we discard type  $t_i$  for  $x$ .

In the following we develop two variants of this approach: a “hard” ILP with rigorous disjointness constraints, and a “soft” ILP which considers type correlations.

**“Hard” ILP with Type Disjointness Constraints.** We infer type disjointness constraints from the YAGO2 knowledge base using occurrence statistics. Types with no overlap in entities or insignificant overlap below a specified threshold are considered disjoint. Notice that this introduces *hard constraints* whereby selecting one type of a disjoint pair rules out the second type. We define type disjointness constraints  $T_i + T_j \leq 1$  for all disjoint pairs  $t_i, t_j$  (e.g. person-artifact, movie-book, city-country, etc.). The ILP is defined as follows:

#### objective

$$\max \sum_i T_i \times w_i$$

#### type disjointness constraint

$$\forall (t_i, t_j)_{\text{disjoint}} T_i + T_j \leq 1$$

The weights  $w_i$  are the aggregated likelihoods as specified in Definition 8.

**“Soft” ILP with Type Correlations.** In many cases, two types are not really mutually exclusive in the strict sense, but the likelihood that an entity belongs to both types is very low. For example, few drummers are also singers. Conversely, certain type combinations are boosted if they are strongly correlated. An example is guitar players and electric guitar players. Our second ILP considers such soft constraints. To this end, we precompute *Pearson correlation* coefficients for all type pairs  $(t_i, t_j)$  based on co-occurrences of types for the same entities. These values  $v_{ij} \in [-1, 1]$  are used as weights in the objective function of the ILP. We additionally introduce pair-wise decision variables  $Y_{ij}$ , set to 1 if the entity at hand belongs to both types  $t_i$  and  $t_j$ , and 0 otherwise. This coupling between the  $Y_{ij}$  variables and the  $T_i, T_j$  variables is enforced by specific constraints. For the objective function, we choose a linear combination of per-type evidence, using weights  $w_i$  as before, and the type-compatibility measure, using weights  $v_{ij}$ . The ILP with correlations is defined as follows:

#### objective

$$\max \alpha \sum_i T_i \times w_i + (1 - \alpha) \sum_{ij} Y_{ij} \times v_{ij}$$

#### type correlation constraints

$$\forall_{i,j} Y_{ij} + 1 \geq T_i + T_j$$

$$\forall_{i,j} Y_{ij} \leq T_i$$

$$\forall_{i,j} Y_{ij} \leq T_j$$

Note that both ILP variants need to be solved per entity, not over all entities together. The “soft” ILP has a size quadratic in the number of candidate types, but this is still a tractable input for modern solvers. We use the Gurobi software package to compute the solutions for the ILP’s. With this design, PEARL can efficiently handle a typical news article in less than a second, and is well geared for keeping up with high-rate content streams in real time. For both the “hard” and “soft” variants of the ILP, the solution is the best types for entity  $x$  satisfying the constraints.

## 5 Evaluation

To define a suitable corpus of test data, we obtained a stream of news documents by subscribing to *Google News* RSS feeds for a few topics over a six-month period (April 2012 – September 2012). This produced 318,434 documents. The topics we subscribed to are: *Angela Merkel, Barack Obama, Business, Entertainment, Hillary Clinton, Joe Biden, Mitt Romney, Newt Gingrich, Rick Santorum, SciTech and Top News*. All our experiments were carried out on this data. The type system used is that of YAGO2, which is derived from WordNet. Human evaluations were carried out on Amazon Mechanical Turk (MTurk), which is a platform for crowd-sourcing tasks that require human input. Tasks on MTurk are small questionnaires consisting of a description and a set of questions.

**Baselines.** We compared PEARL against two state-of-the-art baselines: **i). NNPLB** (No Noun Phrase Left Behind), is the method presented in (Lin 2012), based on the propagation of types for known entities through salient patterns occurring with both known and unknown entities. We implemented the algorithm in (Lin 2012) in our framework, using the relational patterns of PATTY (Nakashole 2012) for comparability. For assessment we sampled from the top-5 highest ranked types for each entity. In our experiments, our implementation of NNPLB achieved precision values comparable to those reported in (Lin 2012). **ii). HYENA** (Hierarchical tYpe classification for Entity NAmes), the method of (Yosef 2012), based on a feature-rich classifier for fine-grained, hierarchical type tagging. This is a state-of-the-art representative of similar methods such as (Rahman 2010; Ling 2012).

**Evaluation Task.** To evaluate the quality of types assigned to emerging entities, we presented turkers with sentences from the news tagged with out-of-KB entities and the types inferred by the methods under test. The turkers task was to assess the correctness of types assigned to an entity mention. To make it easy to understand the task for the turkers, we combined the extracted entity and type into a sentence. For example if PEARL inferred that Brussels Summit is an political event, we generate and present the sentence: *Brussels Summit is an event*. We allowed four possible assessment val-

ues: *a) Very good* output corresponds to a perfect result. *b) Good* output exhibits minor errors. For instance, the description *G20 Summit is an organization* is wrong, because the summit is an event, but G20 is indeed an organization. The problem in this example is incorrect segmentation of a named entity. *c) Wrong* for incorrect types (e.g., *Brussels Summit is a politician*). *d) Not sure / do not know* for other cases.

**Comparing PEARL to Baselines.** Per method, turkers evaluated 105 entity-type pair test samples. We first sampled among out-of-KB entities that were mentioned frequently in the news corpus: in at least 20 different news articles. Each test sample was given to 3 different turkers for assessment. Since the turkers did not always agree if the type for a sample is good or not, we aggregate their answers. We use voting to decide whether the type was assigned correctly to an entity. We consider the following voting variants: i) majority “very good” or “good”, a conservative notion of precision:  $\text{precision}_{lower}$ . ii) at least one “very good” or “good”, a liberal notion of precision:  $\text{precision}_{upper}$ . Table 1 shows precision for PEARL-hard, PEARL-soft, NNPLB, and HYENA, with a 0.9-confidence Wilson score interval (Brown 2001). PEARL-hard outperformed PEARL-soft and also both baselines. HYENA’s relatively poor performance can be attributed to the fact that its features are mainly syntactic such as bi-grams and part-of-speech tags. Web data is challenging, it has a lot of variations in syntactic formulations. This introduces a fair amount of ambiguity which can easily mislead syntactic features. Leveraging semantic features as done by PEARL could improve HYENA’s performance. While the NNPLB method performs better than HYENA, in comparison to PEARL-hard, there is room for improvement. Like HYENA, NNPLB assigns negatively correlated types to the same entity. This limitation could be addressed by applying PEARL’s ILPs and probabilistic weights to the candidate types suggested by NNPLB.

To compute inter-judge agreement we calculated Fleiss’ kappa and Cohen’s kappa  $\kappa$ , which are standard measures. The usual assumption for Fleiss’  $\kappa$  is that labels are categorical, so that each disagreement counts the same. This is not the case in our settings, where different labels may indicate partial agreement (“good”, “very good”). There-

	Precision <sub>lower</sub>	Precision <sub>upper</sub>
PEARL-hard	<b>0.77</b> ±0.08	<b>0.88</b> ±0.06
PEARL-soft	0.53±0.09	0.77±0.09
HYENA	0.26±0.08	0.56±0.09
NNPLB	0.46±0.09	0.68±0.09

Table 1: Comparison of PEARL to baselines.

$\kappa$	<i>Fleiss</i>	<i>Cohen</i>
	0.34	0.45

Table 2: Lower bound estimations for inter-judge agreement kappa: Fleiss’  $\kappa$  & adapted Cohen’s  $\kappa$ .

fore the  $\kappa$  values in Table 2 are lower-bound estimates of agreement in our experiments; the “true agreement” seems higher. Nevertheless, the observed Fleiss  $\kappa$  values show that the task was fairly clear to the turkers; values  $> 0.2$  are generally considered as acceptable (Landis 1977). Cohen’s  $\kappa$  is also not directly applicable to our setting. We approximated it by finding pairs of judges who assessed a significant number of the same entity-type pairs.

	Precision <sub>lower</sub>	Precision <sub>upper</sub>
Freq. mentions	0.77±0.08	0.88±0.06
All mentions	0.65±0.09	0.77±0.08

Table 3: PEARL-hard performance on a sample of frequent entities (mention frequency  $\geq 20$ ) and on a sample of entities of all mention frequencies.

**Mention Frequencies.** We also studied PEARL-hard’s performance on entities of different mention frequencies. The results are shown in Table 3. Frequently mentioned entities provide PEARL with more evidence as they potentially occur with more patterns. Therefore, as expected, precision when sampling over all entities drops a bit. For such infrequent entities, PEARL does not have enough evidence for reliable type assignments.

**Variations of PEARL.** To quantify how various aspects of our approach affect performance, we studied a few variations. The first method is the full PEARL-hard. The second method is PEARL with no ILP (denoted No ILP), only using the probabilistic model. The third variation is PEARL without probabilistic weights (denoted Uniform

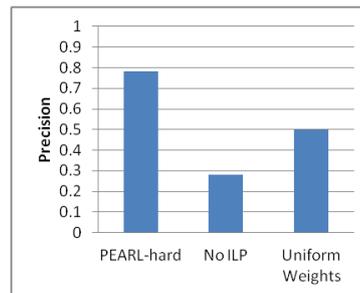


Figure 1: Variations of the PEARL method.

Weights). From Figure 1, it is clear that both the ILP and the weighting model contribute significantly to PEARL’s ability to make precise type assignments. Sample results from PEARL-hard are shown in Table 4.

**NDCG.** For a given entity mention  $e$ , an entity-typing system returns a ranked list of types  $\{t_1, t_2, \dots, t_n\}$ . We evaluated ranking quality using the top-5 ranks for each method. These assessments were aggregated into the normalized discounted cumulative gain (NDCG), a widely used measure for ranking quality. The NDCG values obtained are 0.53, 0.16, and 0.16, for PEARL-hard, HYENA, and NNPLB, respectively. PEARL clearly outperforms the baselines on ranking quality, too.

## 6 Related Work

Tagging mentions of named entities with lexical types has been pursued in previous work. Most well-known is the Stanford named entity recognition (NER) tagger (Finkel 2005) which assigns coarse-grained types like person, organization, location, and other to noun phrases that are likely to denote entities. There is fairly little work on fine-grained typing, notable results being (Fleischmann 2002; Rahman 2010; Ling 2012; Yosef 2012). These methods consider type taxonomies similar to the one used for PEARL, consisting of several hundreds of fine-grained types. All methods use trained classifiers over a variety of linguistic features, most importantly, words and bigrams with part-of-speech tags in a mention and in the textual context preceding and following the mention. In addition, the method of (Yosef 2012) (HYENA) utilizes a big gazetteer of per-type words that occur in Wikipedia anchor texts. This method outperforms earlier techniques on a variety of test

Entity	Inferred Type	Sample Source Sentence (s)
Lochte	medalist	<b>Lochte</b> won America’s lone gold ...
Malick	director	... the red carpet in Cannes for <b>Malick</b> ’s 2011 movie ...
Bonamassa	musician	<b>Bonamassa</b> recorded Driving Towards the Daylight in Las Vegas ... ... <b>Bonamassa</b> opened for B.B. King in Rochester , N.Y.
Analog Man	album	Analog Man is Joe Walsh’s first solo album in 20 years.
Melinda Liu	journalist	... in a telephone interview with journalist <b>Melinda Liu</b> of the Daily Beast.
RealtyTrac	publication	Earlier this month, <b>RealtyTrac</b> reported that ...

Table 4: Sample types inferred by PEARL.

cases; hence it served as one of our baselines.

Closely related to our work is the recent approach of (Lin 2012) (NNPLB) for predicting types for out-of-KB entities. Noun phrases in the subject role in a large collection of fact triples are heuristically linked to Freebase entities. This yields type information for the linked mentions. For unlinkable entities the NNPLB method (inspired by (Kozareva 2011)) picks types based on co-occurrence with salient relational patterns by propagating types of linked entities to unlinkable entities that occur with the same patterns. Unlike PEARL, NNPLB does not attempt to resolve inconsistencies among the predicted types. In contrast, PEARL uses an ILP with type disjointness and correlation constraints to solve and penalize such inconsistencies. NNPLB uses untyped patterns, whereas PEARL harnesses patterns with type signatures. Furthermore, PEARL computes weights for candidate types based on patterns and type signatures. Weight computations in NNPLB are only based on patterns. NNPLB only assigns types to entities that appear in the subject role of a pattern. This means that entities in the object role are not typed at all. In contrast, PEARL infers types for entities in both the subject and object role.

Type disjointness constraints have been studied for other tasks in information extraction (Carlson 2010; Suchanek 2009), but using different formulations.

## 7 Conclusion

This paper addressed the problem of detecting and semantically typing newly emerging entities, to support the life-cycle of large knowledge bases. Our solution, PEARL, draws on a collection of semantically typed patterns for binary relations. PEARL feeds probabilistic evidence derived from

occurrences of such patterns into two kinds of ILPs, considering type disjointness or type correlations. This leads to highly accurate type predictions, significantly better than previous methods, as our crowdsourcing-based evaluation showed.

## References

- S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z.G. Ives: DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International Semantic Web Conference (ISWC)*, pages 722–735, Busan, Korea, 2007.
- M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, O. Etzioni: Open Information Extraction from the Web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2670–2676, Hyderabad, India, 2007.
- K. D. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor: Freebase: a Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages, 1247-1250, Vancouver, BC, Canada, 2008.
- Lawrence D. Brown, T.Tony Cai, Anirban Dasgupta: Interval Estimation for a Binomial Proportion. *Statistical Science* 16: pages 101–133, 2001.
- R. C. Bunescu, M. Pasca: Using Encyclopedic Knowledge for Named entity Disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Trento, Italy, 2006.
- A. Carlson, J. Betteridge, R.C. Wang, E.R. Hruschka, T.M. Mitchell: Coupled Semi-supervised Learning for Information Extraction. In *Proceedings of the Third International Conference on Web Search and Web Data Mining (WSDM)*, pages 101–110, New York, NY, USA, 2010.
- S. Cucerzan: Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-)*

- CoNLL), pages 708–716, Prague, Czech Republic, 2007.
- A. Fader, S. Soderland, O. Etzioni: Identifying Relations for Open Information Extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1535–1545, Edinburgh, UK, 2011.
- J.R. Finkel, T. Grenager, C. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 363–370, Ann Arbor, Michigan, 2005.
- Michael Fleischman, Eduard H. Hovy: Fine Grained Classification of Named Entities. In *Proceedings the International Conference on Computational Linguistics, COLING 2002*.
- X. Han, J. Zhao: Named Entity Disambiguation by Leveraging Wikipedia Semantic Knowledge. In *Proceedings of 18th ACM Conference on Information and Knowledge Management (CIKM)*, pages 215 – 224, Hong Kong, China, 2009.
- C. Havasi, R. Speer, J. Alonso. ConceptNet 3: a Flexible, Multilingual Semantic Network for Common Sense Knowledge. In *Proceedings of the Recent Advances in Natural Language Processing (RANLP)*, Borovets, Bulgaria, 2007.
- Sebastian Hellmann, Claus Stadler, Jens Lehmann, Sren Auer: DBpedia Live Extraction. OTM Conferences (2) 2009: 1209-1223.
- J. Hoffart, M. A. Yosef, I. Bordino and H. Fuerstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, Gerhard Weikum: Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 782–792, Edinburgh, UK, 2011.
- J. Hoffart, F. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo, G. Weikum: YAGO2: Exploring and Querying World Knowledge in Time, Space, Context, and Many Languages. In *Proceedings of the 20th International Conference on World Wide Web (WWW)*, pages 229–232, Hyderabad, India. 2011.
- J. Hoffart, F. Suchanek, K. Berberich, G. Weikum: YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. Artificial Intelligence 2012.
- Z. Kozareva, L. Voevodski, S.-H. Teng: Class Label Enhancement via Related Instances. EMNLP 2011: 118-128
- J. R. Landis, G. G. Koch: The measurement of observer agreement for categorical data in Biometrics. Vol. 33, pp. 159174, 1977.
- C. Lee, Y-G. Hwang, M.-G. Jang: Fine-grained Named Entity Recognition and Relation Extraction for Question Answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 799–800, Amsterdam, The Netherlands, 2007.
- T. Lin, Mausam, O. Etzioni: No Noun Phrase Left Behind: Detecting and Typing Unlinkable Entities. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 893–903, Jeju, South Korea, 2012.
- Xiao Ling, Daniel S. Weld: Fine-Grained Entity Recognition. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2012
- D. N. Milne, I. H. Witten: Learning to Link with Wikipedia. In *Proceedings of 17th ACM Conference on Information and Knowledge Management (CIKM)*, pages 509-518, Napa Valley, California, USA, 2008.
- N. Nakashole, G. Weikum, F. Suchanek: PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1135 - 1145, Jeju, South Korea, 2012.
- V. Nastase, M. Strube, B. Boerschinger, Căcilia Zirn, Anas Elghafari: WikiNet: A Very Large Scale Multi-Lingual Concept Network. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, Malta, 2010.
- H. T. Nguyen, T. H. Cao: Named Entity Disambiguation on an Ontology Enriched by Wikipedia. In *Proceedings of the IEEE International Conference on Research, Innovation and Vision for the Future in Computing & Communication Technologies (RIVF)*, pages 247–254, Ho Chi Minh City, Vietnam, 2008.
- Feng Niu, Ce Zhang, Christopher Re, Jude W. Shavlik: DeepDive: Web-scale Knowledge-base Construction using Statistical Learning and Inference. In the VLDS Workshop, pages 25-28, 2012.
- A. Rahman, Vincent Ng: Inducing Fine-Grained Semantic Classes via Hierarchical and Collective Classification. In *Proceedings the International Conference on Computational Linguistics (COLING)*, pages 931-939, 2010.
- F. M. Suchanek, G. Kasneci, G. Weikum: Yago: a Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web (WWW)* pages, 697-706, Banff, Alberta, Canada, 2007.

- F. M. Suchanek, M. Sozio, G. Weikum: SOFIE: A Self-organizing Framework for Information Extraction. In *Proceedings of the 18th International Conference on World Wide Web (WWW)*, pages 631–640, Madrid, Spain, 2009.
- P.P. Talukdar, F. Pereira: Experiments in Graph-Based Semi-Supervised Learning Methods for Class-Instance Acquisition. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1473-1481, 2010.
- P. Venetis, A. Halevy, J. Madhavan, M. Pasca, W. Shen, F. Wu, G. Miao, C. Wu: Recovering Semantics of Tables on the Web. In *Proceedings of the VLDB Endowment*, PVLDB 4(9), pages, 528–538. 2011.
- W. Wu, H. Li, H. Wang, K. Zhu: Probase: A Probabilistic Taxonomy for Text Understanding. In *Proceedings of the International Conference on Management of Data (SIGMOD)*, pages 481–492, Scottsdale, AZ, USA, 2012.
- M. A. Yosef, S. Bauer, J. Hoffart, M. Spaniol, G. Weikum: HYENA: Hierarchical Type Classification for Entity Names. In *Proceedings the International Conference on Computational Linguistics (COLING)*, to appear, 2012.

# Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction

**Barbara Plank\***

Center for Language Technology  
University of Copenhagen, Denmark  
bplank@gmail.com

**Alessandro Moschitti**

QCRI - Qatar Foundation &  
DISI - University of Trento, Italy  
amoschitti@qf.org.qa

## Abstract

Relation Extraction (RE) is the task of extracting semantic relationships between entities in text. Recent studies on relation extraction are mostly supervised. The clear drawback of supervised methods is the need of training data: labeled data is expensive to obtain, and there is often a mismatch between the training data and the data the system will be applied to. This is the problem of *domain adaptation*. In this paper, we propose to combine (i) term generalization approaches such as word clustering and latent semantic analysis (LSA) and (ii) structured kernels to improve the adaptability of relation extractors to new text genres/domains. The empirical evaluation on ACE 2005 domains shows that a suitable combination of syntax and lexical generalization is very promising for domain adaptation.

## 1 Introduction

Relation extraction is the task of extracting semantic relationships between entities in text, e.g. to detect an employment relationship between the person *Larry Page* and the company *Google* in the following text snippet: *Google CEO Larry Page holds a press announcement at its headquarters in New York on May 21, 2012*. Recent studies on relation extraction have shown that supervised approaches based on either feature or kernel methods achieve state-of-the-art accuracy (Zelenko et al., 2002; Culotta and Sorensen, 2004;

Zhang et al., 2005; Zhou et al., 2005; Zhang et al., 2006; Bunescu, 2007; Nguyen et al., 2009; Chan and Roth, 2010; Sun et al., 2011). However, the clear drawback of supervised methods is the need of training data, which can slow down the delivery of commercial applications in new domains: labeled data is expensive to obtain, and there is often a mismatch between the training data and the data the system will be applied to. Approaches that can cope with domain changes are essential. This is the problem of domain adaptation (DA) or transfer learning (TL). Technically, domain adaptation addresses the problem of learning when the assumption of independent and identically distributed (i.i.d.) samples is violated. Domain adaptation has been studied extensively during the last couple of years for various NLP tasks, e.g. two shared tasks have been organized on domain adaptation for dependency parsing (Nivre et al., 2007; Petrov and McDonald, 2012). Results were mixed, thus it is still a very active research area.

However, to the best of our knowledge, there is almost no work on adapting relation extraction (RE) systems to new domains.<sup>1</sup> There are some prior studies on the related tasks of *multi-task transfer learning* (Xu et al., 2008; Jiang, 2009) and *distant supervision* (Mintz et al., 2009), which are clearly related but different: the former is the problem of how to transfer knowledge from old to new relation types, while distant supervision tries to learn new relations from unlabeled text by exploiting weak-supervision in the form of a knowledge resource (e.g. Freebase). We assume the same relation types but a shift in the underlying

\* The first author was affiliated with the Department of Computer Science and Information Engineering of the University of Trento (Povo, Italy) during the design of the models, experiments and writing of the paper.

<sup>1</sup>Besides an unpublished manuscript of a student project, but it is not clear what data was used. <http://tinyurl.com/bn2hdwk>

data distribution. Weak supervision is a promising approach to improve a relation extraction system, especially to increase its coverage in terms of types of relations covered. In this paper we examine the related issue of changes in the underlying data distribution, while keeping the relations fixed. Even a weakly supervised system is expected to perform well when applied to any kind of text (other domain/genre), thus ideally, we believe that combining domain adaptation with weak supervision is the way to go in the future. This study is a first step towards this.

We focus on *unsupervised* domain adaptation, i.e. no labeled target data. Moreover, we consider a particular domain adaptation setting: *single-system DA*, i.e. learning a single system able to cope with different but related domains. Most studies on DA so far have focused on building a specialized system for every specific target domain, e.g. Blitzer et al. (2006). In contrast, the goal here is to build a single system that can robustly handle several domains, which is in line with the setup of the recent shared task on parsing the web (Petrov and McDonald, 2012). Participants were asked to build a single system that can robustly parse all domains (reviews, weblogs, answers, emails, newsgroups), rather than to build several domain-specific systems. We consider this as a shift in what was considered domain adaptation in the past (adapt from source to a specific target) and what can be considered a somewhat different recent view of DA, that became widespread since 2011/2012. The latter assumes that the target domain(s) is/are not really known in advance. In this setup, the domain adaptation problem boils down to finding a more robust system (Søgaard and Johannsen, 2012), i.e. one wants to build a system that can robustly handle any kind of data.

We propose to combine (i) term generalization approaches and (ii) structured kernels to improve the performance of a relation extractor on new domains. Previous studies have shown that lexical and syntactic features are both very important (Zhang et al., 2006). We combine structural features with lexical information generalized by clusters or similarity. Given the complexity of feature engineering, we exploit kernel methods (Shawe-Taylor and Cristianini, 2004). We encode word clusters or similarity in tree kernels, which, in turn, produce spaces of tree fragments. For example, “president”, “vice-president” and “Texas”,

“US”, are terms indicating an employment relation between a person and a location. Rather than only matching the surface string of words, lexical similarity enables *soft* matches between similar words in convolution tree kernels. In the empirical evaluation on Automatic Content Extraction (ACE) data, we evaluate the impact of convolution tree kernels embedding lexical semantic similarities. The latter is derived in two ways with: (a) Brown word clustering (Brown et al., 1992); and (b) Latent Semantic Analysis (LSA). We first show that our system aligns well with the state of the art on the ACE 2004 benchmark. Then, we test our RE system on the ACE 2005 data, which exploits kernels, structures and similarities for domain adaptation. The results show that combining the huge space of tree fragments generalized at the lexical level provides an effective model for adapting RE systems to new domains.

## 2 Semantic Syntactic Tree Kernels

In kernel-based methods, both learning and classification only depend on the inner product between instances. Kernel functions can be efficiently and implicitly computed by exploiting the dual formulation:  $\sum_{i=1..l} y_i \alpha_i \phi(o_i) \phi(o) + b = 0$ , where  $o_i$  and  $o$  are two objects,  $\phi$  is a mapping from an object to a feature vector  $\vec{x}_i$  and  $\phi(o_i) \phi(o) = K(o_i, o)$  is a kernel function implicitly defining such a mapping. In case of structural kernels,  $K$  determines the shape of the substructures describing the objects. Commonly used kernels in NLP are string kernels (Lodhi et al., 2002) and tree kernels (Moschitti, 2006; Moschitti, 2008).

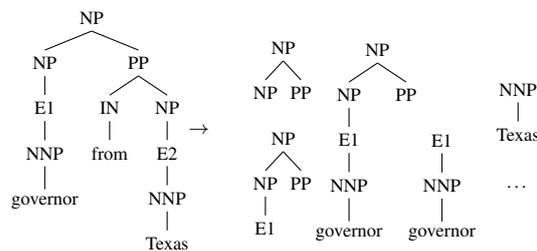


Figure 1: Syntactic tree kernel (STK).

Syntactic tree kernels (Collins and Duffy, 2001) compute the similarity between two trees  $T_1$  and  $T_2$  by counting common sub-trees (cf. Figure 1), without enumerating the whole fragment space. However, if two trees have similar substructures that employ different though related terminal nodes, they will not be matched. This is

clearly a limitation. For instance, the fragments corresponding to `governor` from `Texas` and `head` of `Maryland` are intuitively semantically related and should obtain a higher match when compared to `mother` of `them`.

Semantic syntactic tree kernels (Bloehdorn and Moschitti, 2007a; Bloehdorn and Moschitti, 2007b; Croce et al., 2011) provide one way to address this problem by introducing similarity  $\sigma$  that allows *soft matches* between words and, consequently, between fragments containing them. Let  $N_1$  and  $N_2$  be the set of nodes in  $T_1$  and  $T_2$ , respectively. Moreover, let  $I_i(n)$  be an indicator variable that is 1 if subtree  $i$  is rooted at  $n$  and 0 otherwise. The syntactic semantic convolution kernel  $TK_\sigma$  (Bloehdorn and Moschitti, 2007b) over  $T_1$  and  $T_2$  is computed as  $TK_\sigma(T_1, T_2) = \sum_{n_1 \in N_1, n_2 \in N_2} \Delta_\sigma(n_1, n_2)$  where  $\Delta_\sigma(n_1, n_2) = \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1) I_i(n_2)$  is computed efficiently using the following recursive definition: i) If the nodes  $n_1$  and  $n_2$  are either different or have different number of children then  $\Delta_\sigma(n_1, n_2) = 0$ ; else ii) If  $n_1$  and  $n_2$  are pre-terminals then  $\Delta_\sigma(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} \Delta_\sigma(ch(n_1, j), ch(n_2, j))$ , where  $\sigma$  measures the similarity between the corresponding children of  $n_1$  and  $n_2$ ; iii) If  $n_1$  and  $n_2$  have identical children:  $\Delta_\sigma(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} (1 + \Delta_\sigma(ch(n_1, j), ch(n_2, j)))$ ; else  $\Delta_\sigma(n_1, n_2) = 0$ .  $TK_\sigma$  combines generalized lexical with structural information: it allows matching tree fragments that have the same syntactic structure but differ in their terminals. After introducing related work, we will discuss computational structures for RE and their extension with semantic similarity.

### 3 Related Work

Semantic syntactic tree kernels have been previously used for question classification (Bloehdorn and Moschitti, 2007a; Bloehdorn and Moschitti, 2007b; Croce et al., 2011). These kernels have not yet been studied for either domain adaptation or RE. Brown clusters were studied previously for feature-based approaches to RE (Sun et al., 2011; Chan and Roth, 2010), but they were not yet evaluated in kernels. Thus, we present a novel application of semantic syntactic tree kernels and Brown clusters for domain adaptation of tree-kernel based relation extraction.

Regarding domain adaptation, several methods have been proposed, ranging from instance

weighting (Jiang and Zhai, 2007) to approaches that change the feature representation (Daumé III, 2007) or try to exploit pivot features to find a generalized shared representation between domains (Blitzer et al., 2006). The easy-adapt approach presented in Daumé III (2007) assumes the supervised adaptation setting and is thus not applicable here. Structural correspondence learning (Blitzer et al., 2006) exploits unlabeled data from both source and target domain to find correspondences among features from different domains. These correspondences are then integrated as new features in the labeled data of the source domain. The key to SCL is to exploit pivot features to automatically identify feature correspondences, and as such is applicable to feature-based approaches but not in our case since we do not assume availability of target domain data. Instead, we apply a similar idea where we exploit an entire unlabeled corpus as pivot, and compare our approach to *instance weighting* (Jiang and Zhai, 2007).

Instance weighting is a method for domain adaptation in which instance-dependent weights are assigned to the loss function that is minimized during the training process. Let  $l(x, y, \theta)$  be some loss function. Then, as shown in Jiang and Zhai (2007), the loss function can be weighted by  $\beta_i l(x, y, \theta)$ , such that  $\beta_i = \frac{P_t(x_i)}{P_s(x_i)}$ , where  $P_s$  and  $P_t$  are the source and target distributions, respectively. Huang et al. (2007) present an application of instance weighting to support vector machines by minimizing the following re-weighted function:  $\min_{\theta, \xi} \frac{1}{2} \|\theta\|^2 + C \sum_{i=1}^m \beta_i \xi_i$ . Finding a good weight function is non-trivial (Jiang and Zhai, 2007) and several approximations have been evaluated in the past, e.g. Søgaard and Haulrich (2011) use a bigram-based text classifier to discriminate between domains. We will use a binary classifier trained on RE instance representations.

### 4 Computational Structures for RE

A common way to represent a constituency-based relation instance is the PET (path-enclosed-tree), the smallest subtree including the two target entities (Zhang et al., 2006). This is basically the former structure PAF<sup>2</sup> (predicate argument feature) defined in Moschitti (2004) for the extraction of predicate argument relations. The syntactic rep-

<sup>2</sup>It is the smallest subtree enclosing the predicate and one of its argument node.

resentation used by Zhang et al. (2006) (we will refer to it as PET Zhang) is the PET with enriched entity information: e.g. E1-NAM-PER, including entity type (PER, GPE, LOC, ORG) and mention type (NAM, NOM, PRO, PRE: name, nominal, pronominal or premodifier). An alternative kernel that does not use syntactic information is the Bag-of-Words (BOW) kernel, where a single root node is added above the terminals. Note that in this BOW kernel we actually mark target entities with E1/E2. Therefore, our BOW kernel can be considered an enriched BOW model. If we do not mark target entities, performance drops considerably, as discussed later.

As shown by Zhang et al. (2006), including gold-standard information on entity and mention type substantially improves relation extraction performance. We will use this gold information also in Section 6.1 to show that our system aligns well to the state of the art on the ACE 2004 benchmark. However, in a realistic setting this information is not available or noisy. In fact, as we discuss later, excluding gold entity information decreases system performance considerably. In the case of porting a system to new domains entity information will be unreliable or missing. Therefore, in our domain adaptation experiments on the ACE 2005 data (Section 6.3) we will not rely on this gold information but rather train a system using PET (target mentions only marked with E1/E2 and no gold entity label).<sup>3</sup>

#### 4.1 Syntactic Semantic Structures

Combining syntax with semantics has a clear advantage: it generalizes lexical information encapsulated in syntactic parse trees, while at the same time syntax guides semantics in order to obtain an effective semantic similarity. In fact, lexical information is highly affected by data-sparseness, thus tree kernels combined with semantic information created from additional resources should provide a way to obtain a more robust system.

We exploit this idea here for domain adaptation (DA): if words are generalized by semantic similarity  $LS$ , then in a hypothetical world changing  $LS$  such that it reflects the target domain would

<sup>3</sup>In a setup where gold label info is included, the impact of similarity-based methods is limited – gold information seems to predominate. We argue that whenever gold data is not available, distributional semantics paired with kernels can be useful to improve generalization and complement missing gold info.

allow the system to perform better in the target domain. The question remains how to establish a link between the semantic similarity in the source and target domain. We propose to use an entire unlabeled corpus as pivot: this corpus must be general enough to encapsulate the source and target domains of interest. The idea is to (i) learn semantic similarity between words on the pivot corpus and (ii) use tree kernels embedding such a similarity to learn a RE system on the source, which allows to generalize to the new target domain. This reasoning is related to Structural Correspondence Learning (SCL) (Blitzer et al., 2006). In SCL, a representation shared across domains is learned by exploiting pivot features, where a set of pivot features has to be selected (usually a few thousands). In our case *pivots* are words that co-occur with the target words in a large unlabeled corpus and are thus implicitly represented in the similarity matrix. Thus, in contrast to SCL, we do not need to select a set of pivot features but rather rely on the distributional hypothesis to infer a semantic similarity from a large unlabeled corpus. Then, this similarity is incorporated into the tree kernel that provides the necessary restriction for an effective semantic similarity calculation. One peculiarity of our work is that we exploit a large amount of general data, i.e. data gathered from the web, which is a different but also more challenging scenario than the general unsupervised DA setting where domain specific data is available. We study two ways for term generalization in tree kernels: Brown words clusters and Latent Semantic Analysis (LSA), both briefly described next.

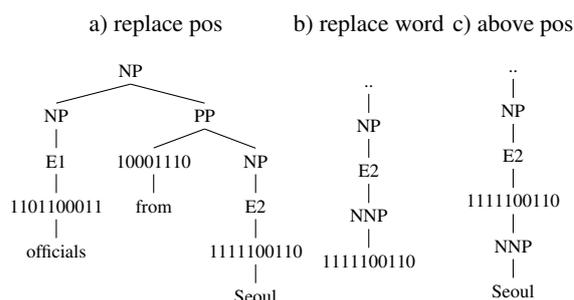


Figure 2: Integrating Brown cluster information

The Brown algorithm (Brown et al., 1992) is a hierarchical agglomerative hard-clustering algorithm. The path from the root of the tree down to a leaf node is represented compactly as a bitstring. By cutting the hierarchy at different levels one can obtain different granularities of word clusters. We

evaluate different ways to integrate cluster information into tree kernels, some of which are illustrated in Figure 2.

For LSA, we compute term similarity functions following the *distributional hypothesis* (Harris, 1964), i.e. the meaning of a word can be described by the set of textual contexts in which it appears. The original word-by-word context matrix  $M$  is decomposed through Singular Value Decomposition (SVD) (Golub and Kahan, 1965), where  $M$  is approximated by  $U_l S_l V_l^T$ . This approximation supplies a way to project a generic term  $w_i$  into the  $l$ -dimensional space using  $W = U_l S_l^{1/2}$ , where each row corresponds to the vectors  $\vec{w}_i$ . Given two words  $w_1$  and  $w_2$ , the term similarity function  $\sigma$  is estimated as the cosine similarity between the corresponding projections  $\vec{w}_1, \vec{w}_2$  and used in the kernel as described in Section 2.

## 5 Experimental Setup

We treat relation extraction as a multi-class classification problem and use SVM-light-TK<sup>4</sup> to train the binary classifiers. The output of the classifiers is combined using the one-vs-all approach. We modified the SVM-light-TK package to include the semantic tree kernels and instance weighting. The entire software package is publicly available.<sup>5</sup> For the SVMs, we use the same parameters as Zhang et al. (2006):  $\lambda = 0.4, c = 2.4$  using the Collins Kernel (Collins and Duffy, 2001). The precision/recall trade-off parameter for the *none* class was found on held-out data:  $j = 0.2$ . Evaluation metrics are standard micro average Precision, Recall and balanced Fscore (F1). To compute statistical significance, we use the approximate randomization test (Noreen, 1989).<sup>6</sup> In all our experiments, we model argument order of the relations explicitly. Thus, for instance for the 7 coarse ACE 2004 relations, we build 14 coarse-grained classifiers (two for each coarse ACE 2004 relation type except for PER-SOC, which is symmetric, and one classifier for the *none* relation).

**Data** We use two datasets. To compare our model against the state of the art we use the *ACE 2004* data. It contains 348 documents and 4,374 positive relation instances. To generate the training data, we follow prior studies and extract an instance for every pair of mentions in the same

sentence, which are separated by no more than three other mentions (Zhang et al., 2006; Sun et al., 2011). After data preprocessing, we obtained 4,327 positive and 39,120 negative instances.

ACE 2005	docs	sents	ASL	relations
nw+bn	298	5029	18.8	3562
bc	52	2267	16.3	1297
cts	34	2696	15.3	603
wl	114	1697	22.6	677

Table 1: Overview of the ACE 2005 data.

For the domain adaptation experiments we use the *ACE 2005* corpus. An overview of the data is given in Table 1. Note that this data is different from ACE 2004: it covers different years (ACE 2004: texts from 2001-2002; ACE 2005: 2003-2005). Moreover, the annotation guidelines have changed (for example, ACE 2005 contains no discourse relation, some relation (sub)types have changed/moved, and care must be taken for differences in SGM markup, etc.).

More importantly, the ACE 2005 corpus covers additional domains: weblogs, telephone conversation, usenet and broadcast conversation. In the experiments, we use news (the union of nw and bn) as source domain, and weblogs (wl), telephone conversations (cts) and broadcast conversation (bc) as target domains.<sup>7</sup> We take half of bc as only target development set, and leave the remaining data and domains for final testing (since they are already small, cf. Table 1). To get a feeling of how these domains differ, Figure 3 depicts the distribution of relations in each domain and Table 2 provides the most frequent out-of-vocabulary words together with their percentage.

**Lexical Similarity and Clustering** We applied LSA to ukWaC (Baroni et al., 2009), a 2 billion word corpus constructed from the Web<sup>8</sup> using the *s-space* toolkit.<sup>9</sup> Dimensionality reduction was performed using SVD with 250 dimensions, following (Croce et al., 2011). The co-occurrence matrix was transformed by *tfidf*. For the Brown word clusters, we used Percy Liang’s implementation<sup>10</sup> of the Brown clustering algorithm (Liang, 2005). We incorporate cluster information by us-

<sup>7</sup>We did not consider the usenet subpart, since it is among the smaller domains and data-preprocessing was difficult.

<sup>8</sup><http://wacky.sslmit.unibo.it/>

<sup>9</sup><http://code.google.com/p/airhead-research/>

<sup>10</sup><https://github.com/percyliang/brown-cluster>

<sup>4</sup><http://disi.unitn.it/moschitti/Tree-Kernel.htm>

<sup>5</sup><http://disi.unitn.it/ikernels/RelationExtraction>

<sup>6</sup><http://www.nlpado.de/~sebastian/software/sigf.shtml>

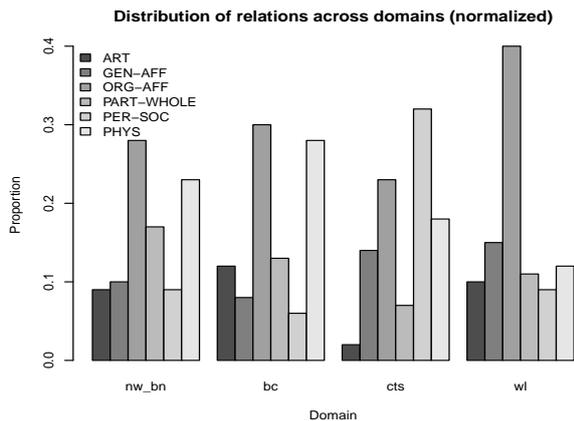


Figure 3: Distribution of relations in ACE 2005.

Dom	Most frequent OOV words
bc (24%)	insurance, unintelligible, malpractice, ph, clip, colonel, crosstalk
cts (34%)	uh, Yeah, um, eh, mhm, uh-huh, ~, ah, mm, th, plo, topic, y, workplace
wl (49%)	title, Starbucks, Well, blog, !!, werkheiser, undefeated, poor, shit

Table 2: For each domain the percentage of target domain words (types) that are unseen in the source together with the most frequent OOV words.

ing the 10-bit cluster prefix (Sun et al., 2011; Chan and Roth, 2010). For the domain adaptation experiments, we use ukWaC corpus-induced clusters as bridge between domains. We limited the vocabulary to that in ACE 2005, which are approximately 16k words. Following previous work, we left case intact in the corpus and induced 1,000 word clusters from words appearing at least 100 times.<sup>11</sup>

**DA baseline** We compare our approach to instance weighting (Jiang and Zhai, 2007). We modified SVM-light-TK such that it takes a parameter vector  $\beta_i, \dots, \beta_m$  as input, where each  $\beta_i$  represents the relative *importance* of example  $i$  with respect to the target domain (Huang et al., 2007; Widmer, 2008). To estimate the importance weights, we train a binary classifier that distinguishes between source and target domain instances. We consider the union of the three target domains as target data. To train the classifier, the source instances are marked as negative and the target instances are marked as positive. Then, this classi-

<sup>11</sup>Clusters are available at <http://disi.unitn.it/ikernels/RelationExtraction>

Prior Work:	Type	P	R	F1
Zhang (2006), tree only	K,yes	74.1	62.4	67.7
Zhang (2006), linear	K,yes	73.5	67.0	70.1
Zhang (2006), poly	K,yes	76.1	68.4	72.1
Sun & Grishman (2011)	F,yes	73.4	67.7	70.4
Jiang & Zhai (2007)	F,no	73.4	70.2	71.3
Our re-implementation:	Type	P	R	F1
Tree only (PET Zhang)	K,yes	70.7	62.5	66.3
Linear composite	K,yes	71.3	66.6	68.9
Polynomial composite	K,yes	72.6	67.7	70.1

Table 3: Comparison to previous work on the 7 relations of ACE 2004. K: kernel-based; F: feature-based; yes/no: models argument order explicitly.

fier is applied to the source data. To obtain the weights  $\beta_i$ , we convert the SVM scores into posterior probabilities by training a sigmoid using the modified Platt algorithm (Lin et al., 2007).<sup>12</sup>

## 6 Results

### 6.1 Alignment to Prior Work

Although most prior studies performed 5-fold cross-validation on ACE 2004, it is often not clear whether the partitioning has been done on the instance or on the document level. Moreover, it is often not stated whether argument order is modeled explicitly, making it difficult to compare system performance. Citing Wang (2008), “We feel that there is a sense of increasing confusion down this line of research”. To ease comparison for future research we use the same 5-fold split on the document level as Sun et al. (2011)<sup>13</sup> and make our system publicly available (see Section 5).

Table 3 shows that our system (bottom) aligns well with the state of the art. Our best system (composite kernel with polynomial expansion) reaches an F1 of 70.1, which aligns well to the 70.4 of Sun et al. (2011) that use the same data-split. This is slightly behind that of Zhang (2006); the reason might be threefold: i) different data partitioning; ii) different pre-processing; iii) they incorporate features from additional sources, i.e. a phrase chunker, dependency parser and semantic resources (Zhou et al., 2005) (we have on average 9 features/instance, they use 40). Since we focus on evaluating the impact of semantic similarity in tree kernels, we think our system is very competitive. Removing gold entity and mention

<sup>12</sup>Other weightings/normalizations (like LDA) didn’t improve the results; best was to take the posteriors and add  $c$ .

<sup>13</sup>[http://cs.nyu.edu/~asun/pub/ACL11\\_CVFileList.txt](http://cs.nyu.edu/~asun/pub/ACL11_CVFileList.txt)

information results in a significant F1 drop from 66.3% to 54.2%. However, in a realistic setting we do not have gold entity info available, especially not in the case when we apply the system to *any kind of text*. Thus, in the domain adaptation setup we assume entity boundaries given but not their label. Clearly, evaluating the approach on predicted mentions, e.g. Giuliano et al. (2007), is another important dimension, however, out of the scope of the current paper.

## 6.2 Tree Kernels with Brown Word Clusters

To evaluate the effectiveness of Brown word clusters in tree kernels, we evaluated different instance representations (cf. Figure 2) on the ACE 2005 development set. Table 4 shows the results.

bc-dev	P	R	F1
baseline	52.2	41.7	46.4
replace word	49.7	38.6	43.4
replace pos	56.3	41.9	<b>48.0</b>
replace pos only mentions	55.3	41.6	47.5
above word	54.5	42.2	47.6
above pos	55.8	41.1	47.3

Table 4: Brown clusters in tree kernels (cf. Fig 2).

To summarize, we found: i) it is generally a bad idea to dismiss lexical information completely, i.e. replacing or ignoring terminals harms performance; ii) the best way to incorporate Brown clusters is to replace the Pos tag with the cluster bitstring; iii) marking all words is generally better than only mentions; this is in contrast to Sun et al. (2011) who found that in their feature-based system it was better to add cluster information to entity mentions only. As we will discuss, the combination of syntax and semantics exploited in this novel kernel avoids the necessity of restricting cluster information to mentions only.

## 6.3 Semantic Tree Kernels for DA

To evaluate the effectiveness of the proposed kernels across domains, we use the ACE 2005 data as testbed. Following standard practices on ACE 2004, the newswire (nw) and broadcast news (bn) data from ACE 2005 are considered training data (labeled source domain). The test data consists of three targets: broadcast conversation, telephone conversation, weblogs. As we want to build a single system that is able to handle heterogeneous data, we do not assume that there is further unlabeled

domain-specific data, but we assume to have a large unlabeled corpus (ukWaC) at our disposal to improve the generalizability of our models.

Table 5 presents the results. In the first three rows we see the performance of the baseline models (PET, BOW and BOW without marking). In-domain (col 1): when evaluated on the same domain the system was trained on (nw+bn, 5-fold cross-validation). Out-of-domain performance (cols 2-4): the system evaluated on the targets, namely broadcast conversation (bc), telephone conversation (cts) and weblogs (wl). While the system achieves a performance of 46.0 F1 within its own domain, the performance drops to 45.3, 43.4 and 34.0 F1 on the target domains, respectively. The BOW kernel that disregards syntax is often less effective (row 2). We see also the effect of target entity marking: the BOW kernel without entity marking performs substantially worse (row 3). For the remaining experiments we use the BOW kernel with entity marking.

Rows 4 and 5 of Table 5 show the effect of using instance weighting for the PET baseline. Two models are shown: they differ in whether PET or BOW was used as instance representation for training the discriminative classifier. Instance weighting shows mixed results: it helps slightly on the weblogs domain, but does not help on broadcast conversation and telephone conversations. Interestingly, the two models used to obtain the weights perform similarly, despite the fact that their performance differs (F1: 70.5 BOW, 73.5 PET); it turns out that the correlation between the weights is high (+0.82).

The next part (rows 6-9) shows the effect of enriching the syntactic structures with either Brown word clusters or LSA. The Brown cluster kernel applied to PET (P\_WC) improves performance over the baseline over *all* target domains. The same holds also for the lexical semantic kernel based on LSA (P\_LSA), however, to only two out of three domains. This suggests that the two kernels capture different information and a combined kernel might be effective. More importantly, the table shows the effect of adding Brown clusters or LSA semantics to the BOW kernel: it can actually hurt performance, sometimes to a small but other times to a considerably degree. For instance, WC applied to PET achieves an F1 of 47.0 (baseline: 45.3) on the bc domain, while applied to BOW it hurts performance significantly, i.e. it drops from

<b>Baseline:</b>	<b>nw+bn (in-dom.)</b>			<b>bc</b>			<b>cts</b>			<b>wl</b>		
	P:	R:	F1:	P:	R:	F1:	P:	R:	F1:	P:	R:	F1:
PET	50.6	42.1	46.0	51.2	40.6	45.3	51.0	37.8	43.4	35.4	32.8	34.0
BOW	55.1	37.3	44.5	57.2	37.1	45.0	57.5	31.8	41.0	41.1	27.2	32.7
BOW no marking	49.6	34.6	40.7	51.5	34.7	41.4	54.6	30.7	39.3	37.6	25.7	30.6
<b>PET adapted:</b>	P:	R:	F:	P:	R:	F:	P:	R:	F:	P:	R:	F:
IW1 (using PET)	51.4	44.1	47.4	49.1	41.1	44.7	50.8	37.5	43.1	35.5	33.9	34.7
IW2 (using BOW)	51.2	43.6	47.1	49.1	41.3	44.9	51.2	37.8	43.5	35.6	33.8	34.7
<b>With Similarity:</b>	P:	R:	F1:	P:	R:	F1:	P:	R:	F1:	P:	R:	F1:
P_WC	55.4	44.6	49.4	54.3	41.4	47.0	55.9	37.1	44.6	40.0	32.7	36.0
B_WC	47.9	36.4	41.4	49.5	35.2	41.2	53.3	33.2	40.9	31.7	24.1	27.4
P_LSA	52.3	44.1	47.9	51.4	41.7	46.0	49.7	36.5	42.1	38.1	36.5	37.3
B_LSA	53.7	37.8	44.4	55.1	33.8	41.9	54.9	32.3	40.7	39.2	28.6	33.0
P+P_WC	55.0	46.5	50.4	54.4	43.4	48.3	54.1	38.1	44.7	38.4	34.5	36.3
P+P_LSA	52.7	46.6	49.5	53.9	45.2	49.2	49.9	37.6	42.9	37.9	38.3	38.1
P+P_WC+P_LSA	55.1	45.9	50.1	55.3	43.1	48.5†	53.1	37.0	43.6	39.9	35.8	37.8†

Table 5: In-domain (first column) and out-of-domain performance (columns two to four) on ACE 2005. PET and BOW are abbreviated by P and B, respectively. If not specified BOW is *marked*.

45.0 to 41.2. This is also the case for LSA applied to the BOW kernel, which drops to 41.9. On the cts domain this is less pronounced. Only on the weblogs domain B\_LSA achieves a minor improvement (from 32.7 to 33.0). In general, distributional semantics constrained by syntax (i.e. combined with PET) can be effectively exploited, while if applied ‘blindly’ – without the guide of syntax (i.e. BOW) – performance might drop, often considerably. We believe that the semantic information does not help the BOW kernel as there is no syntactic information that constrains the application of the noisy source, as opposed to the case with the PET kernel.

As the two semantically enriched kernels, PET\_LSA and PET\_WC, seem to capture different information we use composite kernels (rows 10-11): the baseline kernel (PET) summed with the lexical semantic kernels. As we can see, results improve further: for instance on the bc test set, PET\_WC reaches an F1 of 47.0, while combined with PET (PET+PET\_WC) this improves to 48.3. Adding also PET\_LSA results in the best performance and our final system (last row): the composite kernel (PET+PET\_WC+PET\_LSA) reaches an F1 of 48.5, 43.6 and 37.8 on the target domains, respectively, i.e. with an absolute improvement of: +3.2%, +0.2% and +3.8%, respectively. Two out of three improvements are significant at  $p < 0.05$  (indicated by † in Table 5). Moreover, the system also improved in its own domain (first column),

therefore having achieved robustness.

By performing an error analysis we found that, for instance, the Brown clusters help to generalize locations and professions. For example, the baseline incorrectly considered ‘Dutch filmmaker’ in a PART-WHOLE relation, while our system correctly predicted GEN-AFF(filmmaker,Dutch). ‘Filmmaker’ does not appear in the source, however ‘Dutch citizen’ does. Both ‘citizen’ and ‘filmmaker’ appear in the same cluster, thereby helping the system to recover the correct relation.

<b>Relation:</b>	<b>bc</b>		<b>cts</b>		<b>wl</b>	
	BL	SYS	BL	SYS	BL	SYS
PART-WHOLE	37.8	<b>43.1</b>	<b>59.3</b>	52.3	30.5	<b>36.3</b>
ORG-AFF	60.7	<b>62.9</b>	35.5	<b>42.3</b>	41.0	<b>42.0</b>
PHYS	35.3	<b>37.6</b>	25.4	<b>28.7</b>	25.2	<b>26.9</b>
ART	20.8	<b>37.9</b>	34.5	<b>43.5</b>	26.5	<b>40.3</b>
GEN-AFF	30.1	<b>33.0</b>	16.8	<b>18.6</b>	21.6	<b>28.1</b>
PER-SOC	74.1	<b>74.2</b>	<b>66.3</b>	63.1	42.6	<b>48.0</b>
$\mu$ average	45.3	<b>48.5</b>	43.4	<b>43.6</b>	34.0	<b>37.8</b>

Table 6: F1 per coarse relation type (ACE 2005). SYS is the final model, i.e. last row (PET+PET\_WC+PET\_LSA) of Table 5.

Furthermore, Table 6 provides the performance breakdown per relation for the baseline (BL) and our best system (SYS). The table shows that our system is able to improve F1 on *all* relations for the broadcast and weblogs data. On most relations, this is also the case for the telephone (cts) data, although the overall improvement is not significant. Most errors were made on the PER-SOC

relation, which constitutes the largest portion of cts (cf. Figure 3). As shown in the same figure, the relation distribution of the cts domain is also rather different from the source. This conversation data is a very hard domain, with a lot of disfluencies and spoken language patterns. We believe it is more distant from the other domains, especially from the unlabeled collection, thus other approaches might be more appropriate, e.g. domain identification (Dredze et al., 2010).

## 7 Conclusions and Future Work

We proposed syntactic tree kernels enriched by lexical semantic similarity to tackle the portability of a relation extractor to different domains. The results of diverse kernels exploiting (i) Brown clustering and (ii) LSA show that a suitable combination of syntax and lexical generalization is very promising for domain adaptation. The proposed system is able to improve performance significantly on two out of three target domains (up to 8% relative improvement). We compared it to instance weighting, which gave only modest or no improvements. Brown clusters remained unexplored for kernel-based approaches. We saw that adding cluster information blindly might actually hurt performance. In contrast, adding lexical information combined with syntax can help to improve performance: the syntactic structure enriched with lexical information provides a feature space where syntax constrains lexical similarity obtained from unlabeled data. Thus, semantic syntactic tree kernels appear to be a suitable mechanism to adequately trade off the two kinds of information. In future we plan to extend the evaluation to predicted mentions, which necessarily includes a careful evaluation of pre-processing components, as well as evaluating the approach on other semantic tasks.

## Acknowledgments

We would like to thank Min Zhang for discussions on his prior work as well as the anonymous reviewers for their valuable feedback. The research described in this paper has been supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under the grant #288024: LIMOSINE – Linguistically Motivated Semantic aggregation engiNes.

## References

- Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora. *Language Resources and Evaluation*, pages 209–226.
- John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain Adaptation with Structural Correspondence Learning. In *Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia.
- Stephan Bloehdorn and Alessandro Moschitti. 2007a. Combined syntactic and semantic kernels for text classification. In *ECIR*, pages 307–318.
- Stephan Bloehdorn and Alessandro Moschitti. 2007b. Exploiting Structure and Semantics for Expressive Text Kernels. In *Conference on Information Knowledge and Management*, Lisbon, Portugal.
- Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18:467–479.
- Razvan C. Bunescu. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of ACL*.
- Yee Seng Chan and Dan Roth. 2010. Exploiting background knowledge for relation extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 152–160, Beijing, China, August. Coling 2010 Organizing Committee.
- Michael Collins and Nigel Duffy. 2001. Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems (NIPS 2001)*.
- Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Semantic convolution kernels over dependency trees: smoothed partial tree kernel. In *CIKM*, pages 2013–2016.
- Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on ACL*, Barcelona, Spain.
- Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of ACL*, pages 256–263, Prague, Czech Republic, June.
- Mark Dredze, Tim Oates, and Christine Piatko. 2010. We're not in kansas anymore: Detecting domain changes in streams. In *Proceedings of EMNLP*, pages 585–595, Cambridge, MA.
- Claudio Giuliano, Alberto Lavelli, and Lorenza Romano. 2007. Relation extraction and the influence of automatic named-entity recognition. *ACM Trans. Speech Lang. Process.*, 5(1):2:1–2:26, December.

- G. Golub and W. Kahan. 1965. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, 2(2):pp. 205–224.
- Zellig Harris. 1964. Distributional structure. In Jerrold J. Katz and Jerry A. Fodor, editors, *The Philosophy of Linguistics*. Oxford University Press.
- Jiayuan Huang, Arthur Gretton, Bernhard Schölkopf, Alexander J. Smola, and Karsten M. Borgwardt. 2007. Correcting sample selection bias by unlabeled data. In *In NIPS*. MIT Press.
- Jing Jiang and Chengxiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *In ACL 2007*, pages 264–271.
- Jing Jiang. 2009. Multi-task transfer learning for weakly-supervised relation extraction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th IJCNLP*, pages 1012–1020, Suntec, Singapore.
- Percy Liang. 2005. Semi-Supervised Learning for Natural Language. Master’s thesis, Massachusetts Institute of Technology.
- Hsuan-Tien Lin, Chih-Jen Lin, and Ruby C. Weng. 2007. A note on Platt’s probabilistic outputs for support vector machines. *Mach. Learn.*, 68(3):267–276.
- Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, pages 419–444.
- Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL-IJCNLP*, pages 1003–1011, Suntec, Singapore, August.
- Alessandro Moschitti. 2004. A study on convolution kernels for shallow semantic parsing. In *Proceedings of the 42nd Meeting of the ACL*, Barcelona, Spain.
- Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of the 17th ECML*, Berlin, Germany.
- Alessandro Moschitti. 2008. Kernel methods, syntax and semantics for relational text categorization. In *CIKM*, pages 253–262.
- Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of EMNLP ’09*, pages 1378–1387, Stroudsburg, PA, USA.
- J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 915–932.
- Eric W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley-Interscience.
- Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL).
- John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.
- Anders Søgaard and Martin Haulrich. 2011. Sentence-level instance-weighting for graph-based and transition-based dependency parsing. In *Proceedings of the 12th International Conference on Parsing Technologies, IWPT ’11*, pages 43–47, Stroudsburg, PA, USA.
- Anders Søgaard and Anders Johannsen. 2012. Robust learning in random subspaces: equipping NLP for OOV effects. In *Proceedings of Coling*.
- Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. Semi-supervised relation extraction with large-scale word clustering. In *Proceedings of ACL-HLT*, pages 521–529, Portland, Oregon, USA.
- Mengqiu Wang. 2008. A re-examination of dependency path kernels for relation extraction. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing-IJCNLP*.
- Christian Widmer. 2008. Domain adaptation in sequence analysis. Diplomarbeit, University of Tübingen.
- Feiyu Xu, Hans Uszkoreit, Hond Li, and Niko Felger. 2008. Adaptation of relation extraction rules to new domains. In *Proceedings of LREC’08*, Marrakech, Morocco.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of EMNLP-ACL*, pages 181–201.
- Min Zhang, Jian Su, Danmei Wang, Guodong Zhou, and Chew Lim Tan. 2005. Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In *Proceedings of IJCNLP’2005*, pages 378–389, Jeju Island, South Korea.
- Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of COLING-ACL 2006*, pages 825–832.
- GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting of ACL*, pages 427–434, Ann Arbor, Michigan.

# A joint model of word segmentation and phonological variation for English word-final /t/-deletion

Benjamin Börschinger<sup>1,3</sup> and Mark Johnson<sup>1</sup> and Katherine Demuth<sup>2</sup>

(1) Department of Computing, Macquarie University

(2) Department of Linguistics, Macquarie University

(3) Department of Computational Linguistics, Heidelberg University

{benjamin.borschinger, mark.johnson, katherine.demuth}@mq.edu.au

## Abstract

Word-final /t/-deletion refers to a common phenomenon in spoken English where words such as /wɛst/ “west” are pronounced as [wɛs] “wes” in certain contexts. Phonological variation like this is common in naturally occurring speech. Current computational models of unsupervised word segmentation usually assume idealized input that is devoid of these kinds of variation. We extend a non-parametric model of word segmentation by adding phonological rules that map from underlying forms to surface forms to produce a mathematically well-defined joint model as a first step towards handling variation and segmentation in a single model. We analyse how our model handles /t/-deletion on a large corpus of transcribed speech, and show that the joint model can perform word segmentation and recover underlying /t/s. We find that Bigram dependencies are important for performing well on real data and for learning appropriate deletion probabilities for different contexts.<sup>1</sup>

## 1 Introduction

Computational models of word segmentation try to solve one of the first problems language learners have to face: breaking an unsegmented stream of sound segments into individual words. Currently, most such models assume that the input consists of sequences of phonemes with no pronunciation variation across different occurrences of the same word type. In this paper we describe

<sup>1</sup>The implementation of our model as well as scripts to prepare the data will be made available at <http://web.science.mq.edu.au/~bborschi>. We can't release our version of the Buckeye Corpus (Pitt et al., 2007) directly because of licensing issues.

an extension of the Bayesian models of Goldwater et al. (2009) that incorporates phonological rules to “explain away” surface variation. As a concrete example, we focus on word-final /t/-deletion in English, although our approach is not limited to this case. We choose /t/-deletion because it is a very common and well-studied phenomenon (see Coetzee (2004, Chapter 5) for a review) and segmental deletion is an interesting test-case for our architecture. Recent work has found that /t/-deletion (among other things) is indeed common in child-directed speech (CDS) and, importantly, that its distribution is similar to that in adult-directed speech (ADS) (Dilley et al., to appear). This justifies our using ADS to evaluate our model, as discussed below.

Our experiments are consistent with longstanding and recent findings in linguistics, in particular that /t/-deletion heavily depends on the immediate context and that models ignoring context work poorly on real data. We also examine how well our models identify the probability of /t/-deletion in different contexts. We find that models that capture bigram dependencies between underlying forms provide considerably more accurate estimates of those probabilities than corresponding unigram or “bag of words” models of underlying forms.

In section 2 we discuss related work on handling variation in computational models and on /t/-deletion. Section 3 describes our computational model and section 4 discusses its performance for recovering deleted /t/s. We look at both a situation where word boundaries are pre-specified and only inference for underlying forms has to be performed; and the problem of jointly finding the word boundaries and recovering deleted underlying /t/s. Section 5 discusses our findings, and section 6 concludes with directions for further research.

## 2 Background and related work

The work of Elsner et al. (2012) is most closely related to our goal of building a model that handles variation. They propose a pipe-line architecture involving two separate generative models, one for word-segmentation and one for phonological variation. They model the mapping to surface forms using a probabilistic finite-state transducer. This allows their architecture to handle virtually arbitrary pronunciation variation. However, as they point out, combining the segmentation and the variation model into one joint model is not straight-forward and usual inference procedures are infeasible, which requires the use of several heuristics. We pursue an alternative research strategy here, starting with a single well-studied example of phonological variation. This permits us to develop a joint generative model for both word segmentation and variation which we plan to extend to handle more phenomena in future work.

An earlier work that is close to the spirit of our approach is Naradowsky and Goldwater (2009), who learn spelling rules jointly with a simple stem-suffix model of English verb morphology. Their model, however, doesn't naturally extend to the segmentation of entire utterances.

*/t/-deletion* has received a lot of attention within linguistics, and we point the interested reader to Coetzee (2004, Chapter 5) for a thorough review. Briefly, the phenomenon is as follows: word-final instances of */t/* may undergo deletion in natural speech, such that */west/* “west” is actually pronounced as *[wes]* “wes”.<sup>2</sup> While the frequency of this phenomenon varies across social and dialectal groups, within groups it has been found to be robust, and the probability of deletion depends on its phonological context: a */t/* is more likely to be dropped when followed by a consonant than a vowel or a pause, and it is more likely to be dropped when following a consonant than a vowel as well. We point out two recent publications that are of direct relevance to our research. Dilley et al. (to appear) study word-final variation in stop consonants in CDS, the kind of input we ideally would like to evaluate our models on. They find that “infants largely experience statistical distributions of non-canonical consonantal pronunciation variants [including deletion] that mirror those experienced by adults.” This both directly establishes the need

<sup>2</sup>Following the convention in phonology, we give underlying forms within “/.../” and surface forms within “[...]”.

for computational models to handle this dimension of variation, and justifies our choice of using ADS for evaluation, as mentioned above.

Coetzee and Kawahara (2013) provide a computational study of (among other things) */t/-deletion* within the framework of Harmonic Grammar. They do not aim for a joint model that also handles word segmentation, however, and rather than training their model on an actual corpus, they evaluate on constructed lists of examples, mimicking frequencies of real data. Overall, our findings agree with theirs, in particular that capturing the probability of deletion in different contexts does not automatically result in good performance for recovering individual deleted */t/s*. We will come back to this point in our discussion at the end of the paper.

## 3 The computational model

Our models build on the Unigram and the Bigram model introduced in Goldwater et al. (2009). Figure 1 shows the graphical model for our joint Bigram model (the Unigram case is trivially recovered by generating the  $U_{i,j}$ s directly from  $L$  rather than from  $L_{U_{i,j-1}}$ ). Figure 2 gives the mathematical description of the graphical model and Table 1 provides a key to the variables of our model.

The model generates a latent sequence of *underlying word-tokens*  $U_1, \dots, U_n$ . Each word token is itself a non-empty sequence of segments or phonemes, and each  $U_j$  corresponds to an underlying word form, prior to the application of any phonological rule. This generative process is repeated for each utterance  $i$ , leading to multiple utterances of the form  $U_{i,1}, \dots, U_{i,n_i}$  where  $n_i$  is the number of words in the  $i^{th}$  utterance, and  $U_{i,j}$  is the  $j^{th}$  word in the  $i^{th}$  utterance. Each utterance is padded by an observed utterance boundary symbol  $\$$  to the left and to the right, hence  $U_{i,0} = U_{i,n_i+1} = \$$ .<sup>3</sup> Each  $U_{i,j+1}$  is generated conditionally on its predecessor  $U_{i,j}$  from  $L_{U_{i,j}}$ , as shown in the first row of the lower plate in Figure 1. Each  $L_w$  is a distribution over the possible words that can follow a token of  $w$  and  $L$  is a global distribution over possible words, used as back-off for all  $L_w$ . Just as in Goldwater et al. (2009),  $L$  is drawn from a Dirichlet Process (DP) with base distribution  $B$  and concentration

<sup>3</sup>Each utterance terminates as soon as a  $\$$  is generated, thus determining the number of words  $n_i$  in the  $i^{th}$  utterance. See Goldwater et al. (2009) for discussion.

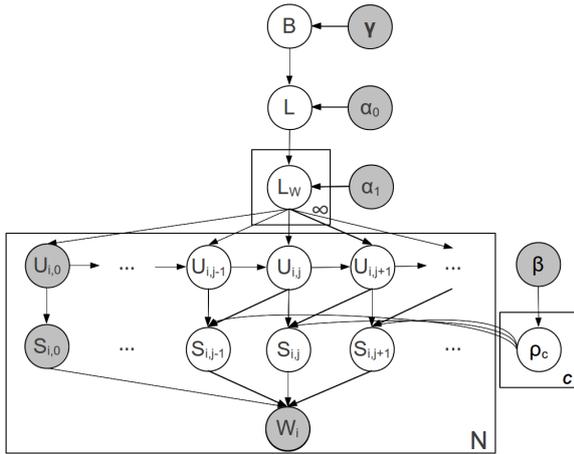


Figure 1: The graphical model for our joint model of word-final /t/-deletion and Bigram word segmentation. The corresponding mathematical description is given in Figure 2. The generative process mimics the intuitively plausible idea of generating underlying forms from some kind of syntactic model (here, a Bigram language model) and then mapping the underlying form to an observed surface-form through the application of a phonological rule component, here represented by the collection of rule probabilities  $\rho_c$ .

parameter  $\alpha_0$ , and the word type specific distributions  $L_w$  are drawn from a  $DP(L, \alpha_1)$ , resulting in a hierarchical DP model (Teh et al., 2006). The base distribution  $B$  functions as a lexical generator, defining a prior distribution over possible words. In principle,  $B$  can incorporate arbitrary prior knowledge about possible words, for example syllable structure (cf. Johnson (2008)). Inspired by Norris et al. (1997), we use a simpler possible word constraint that only rules out sequences that lack a vowel (see Figure 3). While this is clearly a simplification it is a plausible assumption for English data.

Instead of generating the observed sequence of segments  $W$  directly by concatenating the underlying forms as in Goldwater et al. (2009), we map each  $U_{i,j}$  to a corresponding surface-form  $S_{i,j}$  by a probabilistic rule component  $P_R$ . The values over which the  $S_{i,j}$  range are determined by the available phonological processes. In the

$$\begin{aligned}
 L \mid \gamma, \alpha_0 &\sim DP(\alpha_0, B(\cdot \mid \gamma)) \\
 L_w \mid L, \alpha_1 &\sim DP(\alpha_1, L) \\
 \rho_c \mid \beta &\sim Beta(1, 1) \\
 U_{i,0} &= \$ \\
 S_{i,0} &= \$ \\
 U_{i,j+1} \mid U_{i,j}, L_{U_{i,j}} &\sim L_{U_{i,j}} \\
 S_{i,j} \mid U_{i,j}, U_{i,j+1}, \rho &= P_R(\cdot \mid U_{i,j}, U_{i,j+1}) \\
 W_i \mid S_{i,1}, \dots, S_{i,n_i} &= CAT(S_{i,0}, \dots, S_{i,n_i})
 \end{aligned}$$

Figure 2: Mathematical description of our joint Bigram model. The lexical generator  $B(\cdot \mid \gamma)$  is specified in Figure 3 and  $P_R$  is explained in the text below. CAT stands for concatenation without word-boundaries,  $n_i$  refers to the number of words in utterance  $i$ .

Variable	Explanation
$B$	base distribution over possible words
$L$	back-off distribution over words
$L_w$	distribution over words following $w$
$U_{i,j}$	underlying form, a word
$S_{i,j}$	surface realization of $U_{i,j}$ , a word
$\rho_c$	/t/-deletion probability in context $c$
$W_i$	observed segments for $i^{th}$ utterance

Table 1: Key for the variables in Figure 1 and Figure 2. See Figure 3 for the definition of  $B$ .

model we study here, the phonological processes only include a rule for deleting word-final /t/s but in principle,  $P_R$  can be used to encode a wide variety of phonological rules. Here,  $S_{i,j} \in \{U_{i,j}, \text{DELFF}(U_{i,j})\}$  if  $U_{i,j}$  ends in a /t/, and  $S_{i,j} = U_{i,j}$  otherwise, where  $\text{DELFF}(u)$  refers to the same word as  $u$  except that it lacks  $u$ 's final segment.

We look at three kinds of contexts on which a rule's probability of applying depends:

1. a *uniform* context that applies to every word-final position
2. a *right* context that also considers the following segment
3. a *left-right* context that additionally takes the preceding segment into account

For each possible context  $c$  there is a probability  $\rho_c$  which stands for the probability of the rule applying in this context. Writing

$$\gamma \sim \text{Dir}(\langle 0.01, \dots, 0.01 \rangle)$$

$$B(w = x_{1:n} | \gamma) = \begin{cases} \frac{[\prod_{i=1}^n \gamma_{x_i}] \gamma_{\#}}{Z} & \text{if } V(w) \\ 0.0 & \text{if } \neg V(w) \end{cases}$$

Figure 3: Lexical generator with possible word-constraint for words in  $\Sigma^+$ ,  $\Sigma$  being the alphabet of available phonemes.  $x_{1:n}$  is a sequence of elements of  $\Sigma$  of length  $n$ .  $\gamma$  is a probability vector of length  $|\Sigma| + 1$  drawn from a sparse Dirichlet prior, giving the probability for each phoneme and the special word-boundary symbol  $\#$ . The predicate  $V$  holds of all sequences containing at least one vowel.  $Z$  is a normalization constant that adjusts for the mass assigned to the empty and non-possible words.

contexts in the notation familiar from generative phonology (Chomsky and Halle, 1968), our model can be seen as implementing the following rules under the different assumptions:<sup>4</sup>

$$\begin{array}{ll} \text{uniform} & /t/ \rightarrow \emptyset \quad / \text{ \_\_\_\_\_\_}]_{\text{word}} \\ \text{right} & /t/ \rightarrow \emptyset \quad / \text{ \_\_\_\_\_\_}]_{\text{word}} \beta \\ \text{left-right} & /t/ \rightarrow \emptyset \quad / \alpha \text{ \_\_\_\_\_\_}]_{\text{word}} \beta \end{array}$$

We let  $\beta$  range over  $V(\text{owel})$ ,  $C(\text{onsonant})$  and  $\$$  (utterance-boundary), and  $\alpha$  over  $V$  and  $C$ . We define a function  $\text{CONT}$  that maps a pair of adjacent underlying forms  $U_{i,j}, U_{i,j+1}$  to the context of the final segment of  $U_{i,j}$ . For example,  $\text{CONT}(/w\text{est}/, /e\text{v}/)$  returns “ $C \text{ \_\_\_\_\_\_}]_{\text{word}} V$ ” in the *left-right* setting, or simply “ $\text{ \_\_\_\_\_\_}]_{\text{word}}$ ” in the *uniform* setting.  $\text{CONT}$  returns a special  $\text{NOT}$  context if  $U_{i,j}$  doesn’t end in a  $/t/$ . We stipulate that  $\rho_{\text{NOT}} = 0.0$ . Then we can define  $P_R$  as follows:

$$P_R(\text{DELFINAL}(u) | u, r) = \rho_{\text{CONT}(u,r)}$$

$$P_R(u | u, r) = 1 - \rho_{\text{CONT}(u,r)}$$

Depending on the context setting used, our model includes one (*uniform*), three (*right*) or six (*left-right*)  $/t/$ -deletion probabilities  $\rho_c$ . We place a uniform Beta prior on each of those so as to learn their values in the  $\text{LEARN-}\rho$  experiments below.

Finally, the observed unsegmented utterances  $W_i$  are generated by concatenating all  $S_{i,j}$  using the function  $\text{CAT}$ .

We briefly comment on the central intuition of this model, i.e. why it can infer underlying

<sup>4</sup>For *right* there are three and for *left-right* six different rules, one for every instantiation of the context-template.

from surface forms. Bayesian word segmentation models try to compactly represent the observed data in terms of a small set of units (word types) and a short analysis (a small number of word tokens). Phonological rules such as  $/t/$ -deletion can “explain away” an observed surface type such as  $[w\text{es}]$  in terms of the underlying type  $/w\text{est}/$  which is independently needed for surface tokens of  $[w\text{est}]$ . Thus, the  $/t/ \rightarrow \emptyset$  rule makes possible a smaller lexicon for a given number of surface tokens. Obviously, human learners have access to additional cues, such as the meaning of words, knowledge of phonological similarity between segments and so forth. One of the advantages of an explicitly defined generative model such as ours is that it is straight-forward to gradually extend it by adding more cues, as we point out in the discussion.

### 3.1 Inference

Just as for the Goldwater et al. (2009) segmentation models, exact inference is infeasible for our joint model. We extend the collapsed Gibbs breakpoint-sampler described in Goldwater et al. (2009) to perform inference for our extended models. We refer the reader to their paper for additional details such as how to calculate the Bigram probabilities in Figure 4. Here we focus on the required changes to the sampler so as to perform inference under our richer model. We consider the case of a single surface string  $W$ , so we drop the  $i$ -index in the following discussion.

Knowing  $W$ , the problem is to recover the underlying forms  $U_1, \dots, U_n$  and the surface forms  $S_1, \dots, S_n$  for unknown  $n$ . A major insight in Goldwater’s work is that rather than sampling over the latent variables in the model directly (the number of which we don’t even know), we can instead perform Gibbs sampling over a set of boundary variables  $b_1, \dots, b_{|W|-1}$  that jointly determine the values for our variables of interest where  $|W|$  is the length of the surface string  $W$ . For our model, each  $b_j \in \{0, 1, t\}$ , where  $b_j = 0$  indicates absence of a word boundary,  $b_j = 1$  indicates presence of a boundary and  $b_j = t$  indicates presence of a boundary with a preceding underlying  $/t/$ . The relation between the  $b_j$  and the  $S_1, \dots, S_n$  and  $U_1, \dots, U_n$  is illustrated in Figure 5. The required sampling equations are given in Figure 4.

$$P(b_j = 0 \mid \mathbf{b}^{-j}) \propto P(w_{12,u} \mid w_{l,u}, \mathbf{b}^{-j}) \times P_r(w_{12,s} \mid w_{12,u}, w_{r,u}) \times P(w_{r,u} \mid w_{12,u}, \mathbf{b}^{-j} \oplus \langle w_{l,u}, w_{12,u} \rangle) \quad (1)$$

$$P(b_j = t \mid \mathbf{b}^{-j}) \propto P(w_{1,t} \mid w_{l,u}, \mathbf{b}^{-j}) \times P_r(w_{1,s} \mid w_{1,t}, w_{2,u}) \times P(w_{2,u} \mid w_{1,t}, \mathbf{b}^{-j} \oplus \langle w_{l,u}, w_{1,t} \rangle) \\ \times P_r(w_{2,s} \mid w_{2,u}, w_{r,u}) \times P(w_{r,u} \mid w_{2,u}, \mathbf{b}^{-j} \oplus \langle w_{l,u}, w_{1,t} \rangle \oplus \langle w_{1,t}, w_{2,u} \rangle) \quad (2)$$

$$P(b_j = 1 \mid \mathbf{b}^{-j}) \propto P(w_{1,s} \mid w_{l,u}, \mathbf{b}^{-j}) \times P_r(w_{1,s} \mid w_{1,s}, w_{2,u}) \times P(w_{2,u} \mid w_{1,s}, \mathbf{b}^{-j} \oplus \langle w_{l,u}, w_{1,s} \rangle) \\ \times P_r(w_{2,s} \mid w_{2,u}, w_{r,u}) \times P(w_{r,u} \mid w_{2,u}, \mathbf{b}^{-j} \oplus \langle w_{l,u}, w_{1,s} \rangle \oplus \langle w_{1,s}, w_{2,u} \rangle) \quad (3)$$

Figure 4: Sampling equations for our Gibbs sampler, see figure 5 for illustration.  $b_j = 0$  corresponds to no boundary at this position,  $b_j = t$  to a boundary with a preceeding underlying  $/t/$  and  $b_j = 1$  to a boundary with no additional underlying  $/t/$ . We use  $\mathbf{b}^{-j}$  for the statistics determined by all but the  $j^{\text{th}}$  position and  $\mathbf{b}^{-j} \oplus \langle r, l \rangle$  for these statistics plus an additional count of the bigram  $\langle r, l \rangle$ .  $P(w \mid l, \mathbf{b})$  refers to the bigram probability of  $\langle l, w \rangle$  given the the statistics  $\mathbf{b}$ ; we refer the reader to Goldwater et al. (2009) for the details of calculating these bigram probabilities and details about the required statistics for the collapsed sampler.  $P_R$  is defined in the text.

<i>underlying</i>	I	h	i	t	i	t	\$
<i>surface</i>	I	h	i	i	t	\$	
<i>boundaries</i>							
<i>observed</i>	I	h	i	i	t	\$	

Figure 5: The relation between the observed sequence of segments (bottom), the boundary variables  $b_1, \dots, b_{|W|-1}$  the Gibbs sampler operates over (in squares), the latent sequence of surface forms and the latent sequence of underlying forms. When sampling a new value for  $b_3 = t$ , the different word-variables in figure 4 are:  $w_{12,u}=w_{12,s}=hiit$ ,  $w_{1,t}=hit$  and  $w_{1,s}=hi$ ,  $w_{2,u}=w_{2,s}=it$ ,  $w_{l,u}=I$ ,  $w_{r,u}=\$$ . Note that we need a boundary variable at the end of the utterance as there might be an *underlying*  $/t/$  at this position as well. The final boundary variable is set to 1, not  $t$ , because the  $/t/$  in *it* is observed.

## 4 Experiments

### 4.1 The data

We are interested in how well our model handles  $/t/$ -deletion in real data. Ideally, we'd evaluate it on CDS but as of now, we know of no available large enough corpus of accurately hand-transcribed CDS. Instead, we used the Buckeye Corpus (Pitt et al., 2007) for our experiments, a large ADS corpus of interviews with English speakers that have been transcribed with relatively fine phonetic detail, with  $/t/$ -deletion among the things manually annotated. Pointing to the recent work by Dilley et al. (to appear) we want to emphasize that the statistical distribution of  $/t/$ -deletion has been found to be similar for ADS and

orthographic	I don't intend to
transcript	/aɪ r oʊ n i n t ε n d ə/
idealized	/aɪ d oʊ n t i n t ε n d t ə/
t-drop	/aɪ d oʊ n i n t ε n d t ə/

Figure 6: An example fragment from the Buckeye-corpus in orthographic form, the fine transcript available in the Buckeye corpus, a fully idealized pronunciation with canonical dictionary pronunciations and our version of the data with dropped  $/t/$ s.

CDS, at least for read speech.

We automatically derived a corpus of 285,792 word tokens across 48,795 utterances from the Buckeye Corpus by collecting utterances across all interviews and heuristically splitting utterances at speaker-turn changes and indicated silences. The Buckeye corpus lists for each word token a manually transcribed pronunciation in context as well as its canonical pronunciation as given in a pronouncing dictionary. As input to our model, we use the canonical pronunciation unless the pronunciation in context indicates that the final  $/t/$  has been deleted in which case we also delete the final  $/t/$  of the canonical pronunciation Figure 6 shows an example from the Buckeye Corpus, indicating how the original data, a fully idealized version and our derived input that takes into account  $/t/$ -deletions looks like.

Overall,  $/t/$ -deletion is a quite frequent phenomenon with roughly 29% of all underlying  $/t/$ s being dropped. The probabilities become more peaked when looking at finer context; see Table 3 for the empirical distribution of  $/t/$ -dropping for the six different contexts of the *left-right* setting.

## 4.2 Recovering deleted /t/s, given word boundaries

In this set of experiments we are interested in how well our model recovers /t/s when it is provided with the gold word boundaries. This allows us to investigate the strength of the statistical signal for the deletion rule without confounding it with the word segmentation performance, and to see how the different contextual settings *uniform*, *right* and *left-right* handle the data. Concretely, for the example in Figure 6 this means that we tell the model that there are boundaries between /aɪ/, /down/, /ɪntend/, /tu/ and /liv/ but we don’t tell it whether or not these words end in an underlying /t/. Even in this simple example, there are 5 possible positions for the model to posit an underlying /t/. We evaluate the model in terms of F-score, the harmonic mean of recall (the fraction of underlying /t/s the model correctly recovered) and precision (the fraction of underlying /t/s the model predicted that were correct).

In these experiments, we ran a total of 2500 iterations with a burnin of 2000. We collect samples with a lag of 10 for the last 500 iterations and perform *maximum marginal decoding* over these samples (Johnson and Goldwater, 2009), as well as running two chains so as to get an idea of the variance.<sup>5</sup>

We are also interested in how well the model can infer the rule probabilities from the data, that is, whether it can learn values for the different  $\rho_c$  parameters. We compare two settings, one where we perform inference for these parameters assuming a uniform Beta prior on each  $\rho_c$  (LEARN- $\rho$ ) and one where we provide the model with the empirical probabilities for each  $\rho_c$  as estimated off the gold-data (GOLD- $\rho$ ), e.g., for the *uniform* condition 0.29. The results are shown in Table 2.

Best performance for both the Unigram and the Bigram model in the GOLD- $\rho$  condition is achieved under the *left-right* setting, in line with the standard analyses of /t/-deletion as primarily being determined by the preceding and the following context. For the LEARN- $\rho$  condition, the Bigram model still performs best in the *left-right* setting but the Unigram model’s performance drops

<sup>5</sup>As manually setting the hyper-parameters for the DPs in our model proved to be complicated and may be objected to on principled grounds, we perform inference for them under a vague gamma prior, as suggested by Teh et al. (2006) and Johnson and Goldwater (2009), using our own implementation of a slice-sampler (Neal, 2003).

		<i>uniform</i>	<i>right</i>	<i>left-right</i>
Unigram	LEARN- $\rho$	56.52	39.28	23.59
	GOLD- $\rho$	62.08	60.80	66.15
Bigram	LEARN- $\rho$	60.85	62.98	77.76
	GOLD- $\rho$	69.06	69.98	73.45

Table 2: F-score of recovered /t/s with known word boundaries on real data for the three different context settings, averaged over two runs (all standard errors below 2%). Note how the Unigram model always suffers in the LEARN- $\rho$  condition whereas the Bigram model’s performance is actually best for LEARN- $\rho$  in the *left-right* setting.

	C_C	C_V	C_\$	V_C	V_V	V_\$
empirical	0.62	0.42	0.36	0.23	0.15	0.07
Unigram	0.41	0.33	0.17	0.07	0.05	0.00
Bigram	0.70	0.58	0.43	0.17	0.13	0.06

Table 3: Inferred rule-probabilities for different contexts in the *left-right* setting from one of the runs. “C\_C” stands for the context where the deleted /t/ is preceded and followed by a consonant, “V\_\$” stands for the context where it is preceded by a vowel and followed by the utterance boundary. Note how the Unigram model severely under-estimates and the Bigram model slightly over-estimates the probabilities.

in all settings and is now worst in the *left-right* and best in the *uniform* setting.

In fact, comparing the inferred probabilities to the “ground truth” indicates that the Bigram model estimates the true probabilities more accurately than the Unigram model, as illustrated in Table 3 for the *left-right* setting. The Bigram model somewhat overestimates the probability for all post-consonantal contexts but the Unigram model severely underestimates the probability of /t/-deletion across all contexts.

## 4.3 Artificial data experiments

To test our Gibbs sampling inference procedure, we ran it on artificial data generated according to the model itself. If our inference procedure fails to recover the underlying /t/s accurately in this setting, we should not expect it to work well on actual data. We generated our artificial data as follows. We transformed the sequence of canonical pronunciations in the Buckeye corpus (which we take to be underlying forms here) by randomly deleting final /t/s using empirical probabilities as shown in Table 3 to generate a sequence of artificial surface forms that serve as input to our models. We

		uniform	right	left-right
Unigram	LEARN- $\rho$	94.35	23.55 (+)	63.06
	GOLD- $\rho$	94.45	94.20	91.83
Bigram	LEARN- $\rho$	92.72	91.64	88.48
	GOLD- $\rho$	92.88	92.33	89.32

Table 4: F-score of /t/-recovery with known word boundaries on artificial data, each condition tested on data that corresponds to the assumption, averaged over two runs (standard errors less than 2% except (+) = 3.68%).

	Unigram	Bigram
LEARN- $\rho$	33.58	55.64
GOLD- $\rho$	55.92	57.62

Table 5: /t/-recovery F-scores when performing joint word segmentation in the *left-right* setting, averaged over two runs (standard errors less than 2%). See Table 6 for the corresponding segmentation F-scores.

did this for all three context settings, always estimating the deletion probability for each context from the gold-standard. The results of these experiments are given in table 4. Interestingly, performance on these artificial data is considerably better than on the real data. In particular the Bigram model is able to get consistently high F-scores for both the LEARN- $\rho$  and the GOLD- $\rho$  setting. For the Unigram model, we again observe the severe drop in the LEARN- $\rho$  setting for the *right* and *left-right* settings although it does remarkably well in the *uniform* setting, and performs well across all settings in the GOLD- $\rho$  condition. We take this to show that our inference algorithm is in fact working as expected.

#### 4.4 Segmentation experiments

Finally, we are also interested to learn how well we can do word segmentation and underlying /t/-recovery jointly. Again, we look at both the LEARN- $\rho$  and GOLD- $\rho$  conditions but focus on the *left-right* setting as this worked best in the experiments above. For these experiments, we perform simulated annealing throughout the initial 2000 iterations, gradually cooling the temperature from 5 to 1, following the observation by Goldwater et al. (2009) that without annealing, the Bigram model gets stuck in sub-optimal parts of the solution space early on. During the annealing stage, we prevent the model from performing inference

for underlying /t/s so that the annealing stage can be seen as an elaborate initialisation scheme, and we perform joint inference for the remaining 500 iterations, evaluating on the last sample and averaging over two runs. As neither the Unigram nor the Bigram model performs “perfect” word segmentation, we expect to see a degradation in /t/-recovery performance and this is what we find indeed. To give an impression of the impact of /t/-deletion, we also report numbers for running only the segmentation model on the Buckeye data with no deleted /t/s and on the data with deleted /t/s. The /t/-recovery scores are given in Table 5 and segmentation scores in Table 6. Again the Unigram model’s /t/-recovery score degrades dramatically in the LEARN- $\rho$  condition. Looking at the segmentation performance this isn’t too surprising: the Unigram model’s poorer token F-score, the standard measure of segmentation performance on a word token level, suggests that it misses many more boundaries than the Bigram model to begin with and, consequently, can’t recover any potential underlying /t/s at these boundaries. Also note that in the GOLD- $\rho$  condition, our joint Bigram model performs almost as well on data with /t/-deletions as the word segmentation model on data that includes no variation at all.

The generally worse performance of handling variation as measured by /t/-recovery F-score when performing joint segmentation is consistent with the finding of Elsner et al. (2012) who report considerable performance drops for their phonological learner when working with induced boundaries (note, however, that their model does not perform joint inference, rather the induced boundaries are given to their phonological learner as ground-truth).

## 5 Discussion

There are two interesting findings from our experiments. First of all, we find a much larger difference between the Unigram and the Bigram model in the LEARN- $\rho$  condition than in the GOLD- $\rho$  condition. We suggest that this is due to the Unigram model’s lack of dependencies between underlying forms, depriving it of an important source of evidence. Bigram dependencies provide additional evidence for underlying /t/ that are deleted on the surface, and because the Bigram model identifies these underlying /t/ more accurately, it can also estimate the /t/ deletion probability more accurately.

	Unigram	Bigram
LEARN- $\rho$	54.53	72.55 (2.3%)
GOLD- $\rho$	54.51	73.18
NO- $\rho$	54.61	70.12
NO-VAR	54.12	73.99

Table 6: Word segmentation F-scores for the /t/-recovery F-scores in Table 5 averaged over two runs (standard errors less than 2% unless given). NO- $\rho$  are scores for running just the word segmentation model with no /t/-deletion rule on the data that includes /t/-deletion, NO-VAR for running just the word segmentation model on the data with no /t/-deletions.

For example, /t/ dropping in “don’t you” yields surface forms “don you”. Because the word bigram probability  $P(\text{you} \mid \text{don’t})$  is high, the bigram model prefers to analyse surface “don” as underlying “don’t”. The Unigram model does not have access to word bigram information so the underlying forms it posits are less accurate (as shown in Table 2), and hence the estimate of the /t/-deletion probability is also less accurate. When the probabilities of deletion are pre-specified the Unigram model performs better but still considerably worse than the Bigram model when the word boundaries are known, suggesting the importance of non-phonological contextual effects that the Bigram model but not the Unigram model can capture. This suggests that for example word predictability in context might be an important factor contributing to /t/-deletion.

The other striking finding is the considerable drop in performance between running on naturalistic and artificially created data. This suggests that the natural distribution of /t/-deletion is much more complex than can be captured by statistics over the phonological contexts we examined. Following Guy (1991), a finer-grained distinction for the preceding segments might address this problem.

Yet another suggestion comes from the recent work in Coetzee and Kawahara (2013) who claim that “[a] model that accounts perfectly for the overall rate of application of some variable process therefore does not necessarily account very well for the actual application of the process to individual words.” They argue that in particular the extremely high deletion rates typical of high frequency items aren’t accurately captured when the

deletion probability is estimated across all types. A look at the error patterns of our model on a sample from the Bigram model in the LEARN- $\rho$  setting on the naturalistic data suggests that this is in fact a problem. For example, the word “just” has an extremely high rate of deletion with  $\frac{1746}{2442} = 0.71\%$ . While many tokens of “jus” are “explained away” through predicting underlying /t/s, the (literally) extra-ordinary frequency of “jus”-tokens lets our model still posit it as an underlying form, although with a much dampened frequency (of the 1746 surface tokens, 1081 are analysed as being realizations of an underlying “just”).

The /t/-recovery performance drop when performing joint word segmentation isn’t surprising as even the Bigram model doesn’t deliver a very high-quality segmentation to begin with, leading to both sparsity (through missed word-boundaries) and potential noise (through misplaced word-boundaries). Using a more realistic generative process for the underlying forms, for example an Adaptor Grammar (Johnson et al., 2007), could address this shortcoming in future work without changing the overall architecture of the model although novel inference algorithms might be required.

## 6 Conclusion and outlook

We presented a joint model for word segmentation and the learning of phonological rule probabilities from a corpus of transcribed speech. We find that our Bigram model reaches 77% /t/-recovery F-score when run with knowledge of true word-boundaries and when it can make use of both the preceding and the following phonological context, and that unlike the Unigram model it is able to learn the probability of /t/-deletion in different contexts. When performing joint word segmentation on the Buckeye corpus, our Bigram model reaches around above 55% F-score for recovering deleted /t/s with a word segmentation F-score of around 72% which is 2% better than running a Bigram model that does not model /t/-deletion.

We identified additional factors that might help handling /t/-deletion and similar phenomena. A major advantage of our generative model is the ease and transparency with which its assumptions can be modified and extended. For future work we plan to incorporate into our model richer phonological contexts, item- and frequency-specific probabilities and more direct use of word

predictability. We also plan to extend our model to handle additional phenomena, an obvious candidate being /d/-deletion.

Also, the two-level architecture we present is not limited to the mapping being defined in terms of rules rather than constraints in the spirit of Optimality Theory (Prince and Smolensky, 2004); we plan to explore this alternative path as well in future work.

To conclude, we presented a model that provides a clean framework to test the usefulness of different factors for word segmentation and handling phonological variation in a controlled manner.

## Acknowledgements

We thank the anonymous reviewers for their valuable comments. This research was supported under Australian Research Council's Discovery Projects funding scheme (project numbers DP110102506 and DP110102593).

## References

- Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English*. Haper & Row, New York.
- Andries W. Coetzee and Shigeto Kawahara. 2013. Frequency biases in phonological variation. *Natural Language and Linguistic Theory*, 31:47–89.
- Andries W. Coetzee. 2004. *What it Means to be a Loser: Non-Optimal Candidates in Optimality Theory*. Ph.D. thesis, University of Massachusetts , Amherst.
- Laura Dilley, Amanda Millett, J. Devin McAuley, and Tonya R. Bergeson. to appear. Phonetic variation in consonants in infant-directed and adult-directed speech: The case of regressive place assimilation in word-final alveolar stops. *Journal of Child Language*.
- Micha Elsner, Sharon Goldwater, and Jacob Eisenstein. 2012. Bootstrapping a unified model of lexical and phonetic acquisition. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 184–193, Jeju Island, Korea. Association for Computational Linguistics.
- Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.
- Gregory R. Guy. 1991. Contextual conditioning in variable lexical phonology. *Language Variation and Change*, 3(2):223–39.
- Mark Johnson and Sharon Goldwater. 2009. Improving nonparametric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, Boulder, Colorado, June. Association for Computational Linguistics.
- Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. Adaptor Grammars: A framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, Cambridge, MA.
- Mark Johnson. 2008. Using Adaptor Grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, pages 398–406, Columbus, Ohio. Association for Computational Linguistics.
- Jason Naradowsky and Sharon Goldwater. 2009. Improving morphology induction by learning spelling rules. In *Proceedings of the 21st international joint conference on Artificial intelligence*, pages 1531–1536, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Radford M. Neal. 2003. Slice sampling. *Annals of Statistics*, 31:705–767.
- Dennis Norris, James M. Mcqueen, Anne Cutler, and Sally Butterfield. 1997. The possible-word constraint in the segmentation of continuous speech. *Cognitive Psychology*, 34(3):191 – 243.
- Mark A. Pitt, Laura Dilley, Keith Johnson, Scott Kiesling, William Raymond, Elizabeth Hume, and Eric Fosler-Lussier. 2007. Buckeye corpus of conversational speech.
- Alan Prince and Paul Smolensky. 2004. *Optimality Theory: Constraint Interaction in Generative Grammar*. Blackwell.
- Yee Whye Teh, Michael Jordan, Matthew Beal, and David Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581.

# Compositional-ly Derived Representations of Morphologically Complex Words in Distributional Semantics

Angeliki Lazaridou and Marco Marelli and Roberto Zamparelli and Marco Baroni

Center for Mind/Brain Sciences (University of Trento, Italy)

first.last@unitn.it

## Abstract

Speakers of a language can construct an unlimited number of new words through morphological derivation. This is a major cause of data sparseness for corpus-based approaches to lexical semantics, such as distributional semantic models of word meaning. We adapt compositional methods originally developed for phrases to the task of deriving the distributional meaning of morphologically complex words from their parts. Semantic representations constructed in this way beat a strong baseline and can be of higher quality than representations directly constructed from corpus data. Our results constitute a novel evaluation of the proposed composition methods, in which the *full additive* model achieves the best performance, and demonstrate the usefulness of a compositional morphology component in distributional semantics.

## 1 Introduction

Effective ways to represent word meaning are needed in many branches of natural language processing. In the last decades, corpus-based methods have achieved some degree of success in modeling lexical semantics. Distributional semantic models (DSMs) in particular represent the meaning of a word by a vector, the dimensions of which encode corpus-extracted co-occurrence statistics, under the assumption that words that are semantically similar will occur in similar contexts (Turney and Pantel, 2010). Reliable distributional vectors can only be extracted for words that occur in many contexts in the corpus. Not surprisingly, there is a strong correlation between word frequency and vector quality (Bullinaria and Levy, 2007), and since most words occur only once even in very large corpora (Baroni, 2009), DSMs suffer data sparseness.

While word rarity has many sources, one of the most common and systematic ones is the high *productivity* of morphological derivation processes, whereby an unlimited number of new words can be constructed by adding affixes to existing stems (Baayen, 2005; Bauer, 2001; Plag, 1999).<sup>1</sup> For example, in the multi-billion-word corpus we introduce below, perfectly reasonable derived forms such as *lexicalizable* or *affixless* never occur. Even without considering the theoretically infinite number of possible derived nonce words, and restricting ourselves instead to words that are already listed in dictionaries, complex forms cover a high portion of the lexicon. For example, morphologically complex forms account for 55% of the lemmas in the CELEX English database (see Section 4.1 below). In most of these cases (80% according to our corpus) the stem is more frequent than the complex form (e.g., the stem *build* occurs 15 times more often than the derived form *rebuild*, and the latter is certainly not an unusual derived form).

DSMs ignore derivational morphology altogether. Consequently, they cannot provide meaningful representations for new derived forms, nor can they harness the systematic relation existing between stems and derivations (any English speaker can infer that *rebuild* is to *build again*, whether they are familiar with the prefixed form or not) in order to mitigate derived-form sparseness problems. A simple way to handle derivational mor-

---

<sup>1</sup>Morphological *derivation* constructs new words (in the sense of lemmas) from existing lexical items (*resource+ful*→*resourceful*). In this work, we do not treat *inflectional* morphology, pertaining to affixes that encode grammatical features such as number or tense (*dog+s*). We use *morpheme* for any component of a word (*resource* and *-ful* are both morphemes). We use *stem* for the lexical item that constitutes the base of derivation (*resource*) and *affix* (*prefix* or *suffix*) for the element attached to the stem to derive the new form (*-ful*). In English, stems are typically independent words, affixes *bound* morphemes, i.e., they cannot stand alone. Note that a stem can in turn be morphologically derived, e.g., *point+less* in *pointless+ly*. Finally, we use *morphologically complex* as synonymous with *derived*.

phology would be to identify the stem of rare derived words and use its distributional vector as a proxy to derived-form meaning.<sup>2</sup> The meaning of *rebuild* is not that far from that of *build*, so the latter might provide a reasonable surrogate. Still, something is clearly lost (if the author of a text felt the need to use the derived form, the stem was not fully appropriate), and sometimes the jump in meaning can be quite dramatic (*resourceless* and *resource* mean very different things!).

In the past few years there has been much interest in how DSMs can scale up to represent the meaning of larger chunks of text such as phrases or even sentences. Trying to represent the meaning of arbitrarily long constructions by directly collecting co-occurrence statistics is obviously ineffective and thus methods have been developed to derive the meaning of larger constructions as a function of the meaning of their constituents (Baroni and Zamparelli, 2010; Coecke et al., 2010; Mitchell and Lapata, 2008; Mitchell and Lapata, 2010; Socher et al., 2012). *Compositional* distributional semantic models (cDSMs) of word units aim at handling, compositionally, the high productivity of phrases and consequent data sparseness. It is natural to hypothesize that the same methods can be applied to morphology to derive the meaning of complex words from the meaning of their parts: For example, instead of harvesting a *rebuild* vector directly from the corpus, the latter could be constructed from the distributional representations of *re-* and *build*. Besides alleviating data sparseness problems, a system of this sort, that automatically induces the semantic contents of morphological processes, would also be of tremendous theoretical interest, given that the semantics of derivation is a central and challenging topic in linguistic morphology (Dowty, 1979; Lieber, 2004).

In this paper, we explore, for the first time (except for the proof-of-concept study in Guevara (2009)), the application of cDSMs to derivational morphology. We adapt a number of composition methods from the literature to the morphological setting, and we show that some of these methods can provide better distributional representations of derived forms than either those directly harvested from a large corpus, or those obtained by using the stem as a proxy to derived-form meaning. Our

---

<sup>2</sup>Of course, spotting and segmenting complex words is a big research topic unto itself (Beesley and Karttunen, 2000; Black et al., 1991; Sproat, 1992), and one we completely sidestep here.

results suggest that exploiting morphology could improve the quality of DSMs in general, extend the range of tasks that cDSMs can successfully model and support the development of new ways to test their performance.

## 2 Related work

Morphological induction systems use corpus-based methods to decide if two words are morphologically related and/or to segment words into morphemes (Dreyer and Eisner, 2011; Goldsmith, 2001; Goldwater and McClosky, 2005; Goldwater, 2006; Naradowsky and Goldwater, 2009; Wicentowski, 2004). Morphological induction has recently received considerable attention since morphological analysis can mitigate data sparseness in domains such as parsing and machine translation (Goldberg and Tsarfaty, 2008; Lee, 2004). Among the cues that have been exploited there is distributional similarity among morphologically related words (Schone and Jurafsky, 2000; Yarowsky and Wicentowski, 2000). Our work, however, differs substantially from this track of research. We do not aim at segmenting morphological complex words or identifying paradigms. Our goal is to automatically construct, given distributional representations of stems and affixes, semantic representations for the derived words containing those stems and affixes. A morphological induction system, given *rebuild*, will segment it into *re-* and *build* (possibly using distributional similarity between the words as a cue). Our system, given *re-* and *build*, predicts the (distributional semantic) meaning of *rebuild*.

Another emerging line of research uses distributional semantics to model human intuitions about the semantic transparency of morphologically derived or compound expressions and how these impact various lexical processing tasks (Kuperman, 2009; Wang et al., 2012). Although these works exploit vectors representing complex forms, they do not attempt to generate them compositionally.

The only similar study we are aware of is that of Guevara (2009). Guevara found a systematic geometric relation between corpus-based vectors of derived forms sharing an affix and their stems, and used this finding to motivate the composition method we term *lexfunc* below. However, unlike us, he did not test alternative models, and he only presented a qualitative analysis of the trajectories triggered by composition with various affixes.

### 3 Composition methods

Distributional semantic models (DSMs), also known as vector-space models, semantic spaces, or by the names of famous incarnations such as Latent Semantic Analysis or Topic Models, approximate the meaning of words with vectors that record their patterns of co-occurrence with corpus context features (often, other words). There is an extensive literature on how to develop such models and on their evaluation. Recent surveys include Clark (2012), Erk (2012) and Turney and Pantel (2010). We focus here on compositional DSMs (cDSMs). Since the very inception of distributional semantics, there have been attempts to compose meanings for sentences and larger passages (Landauer and Dumais, 1997), but interest in compositional DSMs has skyrocketed in the last few years, particularly since the influential work of Mitchell and Lapata (2008; 2009; 2010). For the current study, we have reimplemented and adapted to the morphological setting all cDSMs we are aware of, excluding the tensor-product-based models that Mitchell and Lapata (2010) have shown to be empirically disappointing and the models of Socher and colleagues (Socher et al., 2011; Socher et al., 2012), that require complex optimization procedures whose adaptation to morphology we leave to future work.

Mitchell and Lapata proposed a set of simple and effective models in which the composed vectors are obtained through component-wise operations on the constituent vectors. Given input vectors  $\mathbf{u}$  and  $\mathbf{v}$ , the multiplicative model (**mult**) returns a composed vector  $\mathbf{c}$  with:  $c_i = u_i v_i$ . In the weighted additive model (**wadd**), the composed vector is a weighted sum of the two input vectors:  $\mathbf{c} = \alpha \mathbf{u} + \beta \mathbf{v}$ , where  $\alpha$  and  $\beta$  are two scalars. In the **dilation** model, the output vector is obtained by first decomposing one of the input vectors, say  $\mathbf{v}$ , into a vector parallel to  $\mathbf{u}$  and an orthogonal vector. Following this, the parallel vector is dilated by a factor  $\lambda$  before re-combining. This results in:  $\mathbf{c} = (\lambda - 1)\langle \mathbf{u}, \mathbf{v} \rangle \mathbf{u} + \langle \mathbf{u}, \mathbf{u} \rangle \mathbf{v}$ .

Guevara (2010) and Zanzotto et al. (2010) propose the full additive model (**fulladd**), where the two vectors to be added are pre-multiplied by weight matrices:  $\mathbf{c} = \mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v}$

Since the Mitchell and Lapata and *fulladd* models were developed for phrase composition, the two input vectors were taken to be, very straightforwardly, the vectors of the two words to be com-

posed into the phrase of interest. In morphological derivation, at least one of the items to be composed (the affix) is a bound morpheme. In our adaptation of these composition models, we build bound morpheme vectors by accumulating the contexts in which a set of derived words containing the relevant morphemes occur, e.g., the *re-* vector aggregates co-occurrences of *redo*, *remake*, *retry*, etc.

Baroni and Zamparelli (2010) and Coecke et al. (2010) take inspiration from formal semantics to characterize composition in terms of function application, where the distributional representation of one element in a composition (the *functor*) is not a vector but a function. Given that linear functions can be expressed by matrices and their application by matrix-by-vector multiplication, in this *lexical function* (**lexfunc**) model, the functor is represented by a matrix  $\mathbf{U}$  to be multiplied with the argument vector  $\mathbf{v}$ :  $\mathbf{c} = \mathbf{U}\mathbf{v}$ . In the case of morphology, it is natural to treat bound affixes as functions over stems, since affixes encode the systematic semantic patterns we intend to capture. Unlike the other composition methods, *lexfunc* does not require the construction of distributional vectors for affixes. A matrix representation for every affix is instead induced directly from examples of stems and the corresponding derived forms, in line with the intuition that every affix corresponds to a different pattern of change of the stem meaning.

Finally, as already discussed in the Introduction, performing no composition at all but using the stem vector as a surrogate of the derived form is a reasonable strategy. We saw that morphologically derived words tend to appear less frequently than their stems, and in many cases the meanings are close. Consequently, we expect a **stem-only** “composition” method to be a strong baseline in the morphological setting.

## 4 Experimental setup

### 4.1 Morphological data

We obtained a list of stem/derived-form pairs from the CELEX English Lexical Database, a widely used 100K-lemma lexicon containing, among other things, information about the derivational structure of words (Baayen et al., 1995). For each derivational affix present in CELEX, we extracted from the database the full list of stem/derived pairs matching its most common part-of-speech signature (e.g., for *-er* we only considered pairs

Affix	Stem/Der. POS	Training Items	HQ/Tot. Test Items	Avg. SDR
-able	verb/adj	177	30/50	5.96
-al	noun/adj	245	41/50	5.88
-er	verb/noun	824	33/50	5.51
-ful	noun/adj	53	42/50	6.11
-ic	noun/adj	280	43/50	5.99
-ion	verb/noun	637	38/50	6.22
-ist	noun/noun	244	38/50	6.16
-ity	adj/noun	372	33/50	6.19
-ize	noun/verb	105	40/50	5.96
-less	noun/adj	122	35/50	3.72
-ly	adj/adv	1847	20/50	6.33
-ment	verb/noun	165	38/50	6.06
-ness	adj/noun	602	33/50	6.29
-ous	noun/adj	157	35/50	5.94
-y	noun/adj	404	27/50	5.25
in-	adj/adj	101	34/50	3.39
re-	verb/verb	86	27/50	5.28
un-	adj/adj	128	36/50	3.23
<i>tot</i>	<i>*/*</i>	6549	623/900	5.52

Table 1: Derivational morphology dataset

having a verbal stem and nominal derived form). Since CELEX was populated by semi-automated morphological analysis, it includes forms that are probably not synchronically related to their stems, such as *crypt+ic* or *re+form*. However, we did not manually intervene on the pairs, since we are interested in training and testing our methods in realistic, noisy conditions. In particular, the need to pre-process corpora to determine which forms are “opaque”, and should thus be bypassed by our systems, would greatly reduce their usefulness. Pairs in which either word occurred less than 20 times in our source corpus (described in Section 4.2 below) were filtered out and, in our final dataset, we only considered the 18 affixes (3 prefixes and 15 suffixes) with at least 100 pairs meeting this condition. We randomly chose 50 stem/derived pairs (900 in total) as test data. The remaining data were used as training items to estimate the parameters of the composition methods. Table 1 summarizes various characteristics of the dataset<sup>3</sup> (the last two columns of the table are explained in the next paragraphs).

**Annotation of quality of test vectors** The quality of the corpus-based vectors representing derived test items was determined by collecting human semantic similarity judgments in a crowdsourcing survey. In particular, we use the similarity of a vector to its nearest neighbors (NNs) as a proxy measure of quality. The underlying assump-

<sup>3</sup>Available from <http://clic.cimec.unitn.it/composes>

tion is that a vector, in order to be a good representation of the meaning of the corresponding word, should lie in a region of semantic space populated by intuitively similar meanings, e.g., we are more likely to have captured the meaning of *car* if the NN of its vector is the *automobile* vector rather than *potato*. Therefore, to measure the quality of a given vector, we can look at the average similarity score provided by humans when comparing this very vector with its own NNs.

All 900 derived vectors from the test set were matched with their three closest NNs in our semantic space (see Section 4.2), thus producing a set of 2,700 word pairs. These pairs were administered to CrowdFlower users,<sup>4</sup> who were asked to judge the relatedness of the two meanings on a 7-point scale (higher for more related). In order to ensure that participants were committed to the task and exclude non-proficient English speakers, we used 60 control pairs as gold standard, consisting of either perfect synonyms or completely unrelated words. We obtained 30 judgments for each derived form (10 judgments for each of 3 neighbor comparisons), with mean participant agreement of 58%. These ratings were averaged item-wise, resulting in a Gaussian distribution with a mean of 3.79 and a standard deviation of 1.31. Finally, each test item was marked as high-quality (HQ) if its derived form received an average score of at least 3, as low-quality (LQ) otherwise. Table 1 reports the proportion of HQ test items for each affix, and Table 2 reports some examples of HQ and LQ items with the corresponding NNs. It is worth observing that the NNs of the LQ items, while not as relevant as the HQ ones, are hardly random.

**Annotation of similarity between stem and derived forms** Derived forms differ in terms of how far their meaning is with respect to that of their stem. Certain morphological processes have systematically more impact than others on meaning: For example, the adjectival prefix *in-* negates the meaning of the stem, whereas *-ly* has the sole function to convert an adjective into an adverb. But the very same affix can affect different stems in different ways. For example, *remelt* means little more than to *melt again*, but *rethink* has subtler implications of changing one’s way to look at a problem, and while one of the senses of *cycling* is present in *recycle*, it takes some effort to see their relation.

<sup>4</sup><http://www.crowdflower.com>

Affix	Type	Derived form	Neighbors
-ist	HQ	transcendentalist	mythologist, futurist, theosophist
	LQ	florist	Harrod, wholesaler, stockist
-ity	HQ	publicity	publicise, press, publicize
	LQ	sparsity	dissimilarity, contiguity, perceptibility
-ment	HQ	advertisement	advert, promotional, advertising
	LQ	inducement	litigant, contractually, voluntarily
in-	HQ	inaccurate	misleading, incorrect, erroneous
	LQ	inoperable	metastasis, colorectal, biopsy
re-	HQ	recapture	retake, besiege, capture
	LQ	rename	defunct, officially, merge

Table 2: Examples of HQ and LQ derived vectors with their NNs

We conducted a separate crowdsourcing study where participants were asked to rate the 900 test stem/derived pairs for the strength of their semantic relationship on a 7-point scale. We followed a procedure similar to the one described for quality measurement; 7 judgments were collected for each pair. Participants’ agreement was at 60%. The last column of Table 1 reports the average stem/derived relatedness (**SDR**) for the various affixes. Note that the affixes with systematically lower SDR are those carrying a negative meaning (*in-*, *un-*, *-less*), whereas those with highest SDR do little more than changing the POS of the stem (*-ion*, *-ly*, *-ness*). Among specific pairs with very low relatedness we encounter *hand/handy*, *bear/bearable* and *active/activist*, whereas *compulsory/compulsorily*, *shameless/shamelessness* and *chaos/chaotic* have high SDR. Since the distribution of the average ratings was negatively skewed (mean rating: 5.52, standard deviation: 1.26),<sup>5</sup> we took 5 as the rating threshold to classify items as having high (**HR**) or low (**LR**) relatedness to their stems.

## 4.2 Distributional semantic space<sup>6</sup>

We use as our source corpus the concatenation of ukWaC, the English Wikipedia (2009 dump) and the BNC,<sup>7</sup> for a total of about 2.8 billion tokens. We collect co-occurrence statistics for the top 20K content words (adjectives, adverbs, nouns, verbs)

<sup>5</sup>The negative skew is not surprising, as derived forms must have some relation to their stems!

<sup>6</sup>Most steps of the semantic space construction and composition pipelines were implemented using the DISSECT toolkit: <https://github.com/composes-toolkit/dissect>.

<sup>7</sup><http://wacky.sslmit.unibo.it>, <http://en.wikipedia.org>, <http://www.natcorp.ox.ac.uk>

in lemma format, plus any item from the morphological dataset described above that was below this rank. The top 20K content words also constitute our context elements. We use a standard bag-of-words approach, counting collocates in a narrow 2-word before-and-after window. We apply (non-negative) Pointwise Mutual Information as weighting scheme and dimensionality reduction by Non-negative Matrix Factorization, setting the number of reduced-space dimensions to 350. These settings are chosen without tuning, and are based on previous experiments where they produced high-quality semantic spaces (Boleda et al., 2013; Bullinaria and Levy, 2007).

## 4.3 Implementation of composition methods

All composition methods except *mult* and *stem* have weights to be estimated (e.g., the  $\lambda$  parameter of *dilation* or the affix matrices of *lexfunc*). We adopt the estimation strategy proposed by Guevara (2010) and Baroni and Zamparelli (2010), namely we pick parameter values that optimize the mapping between stem and derived vectors directly extracted from the corpus. To learn, say, a *lexfunc* matrix representing the prefix *re-*, we extract vectors of  $V/reV$  pairs that occur with sufficient frequency (*visit/revisit*, *think/rethink*...). We then use least-squares methods to find weights for the *re-* matrix that minimize the distance between each *reV* vector generated by the model given the input  $V$  and the corresponding corpus-observed derived vector (e.g., we try to make the model-predicted *re+visit* vector as similar as possible to the corpus-extracted one). This is a general estimation approach that does not require task-specific hand-labeled data, and for which simple analytical solutions of the least-squares error prob-

lem exist for all our composition methods. We use only the training items from Section 4.1 for estimation. Note that, unlike the test items, these have not been annotated for quality, so we are adopting an unsupervised (no manual labeling) but noisy estimation method.<sup>8</sup>

For the *lexfunc* model, we use the training items separately to obtain weight matrices representing each affix, whereas for the other models all training data are used together to globally derive single sets of affix and stem weights. For the *wadd* model, the learning process results in  $0.16 \times \text{affix} + 0.33 \times \text{stem}$ , i.e., the affix contributes only half of its mass to the composition of the derived form. For *dilation*, we stretch the stem (i.e.,  $\mathbf{v}$  of the dilation equation is the stem vector), since it should provide richer contents than the affix to the derived meaning. We found that, on average across the training pairs, *dilation* weighted the stem 20 times more heavily than the affix ( $0.05 \times \text{affix} + 1 \times \text{stem}$ ). We then expect that the *dilation* model will have similar performance to the baseline *stem* model, as confirmed below.<sup>9</sup>

For all methods, vectors were normalized before composing both in training and in generation.

## 5 Experiment 1: approximating high-quality corpus-extracted vectors

The first experiment investigates to what extent composition models can approximate high-quality (HQ) corpus-extracted vectors representing derived forms. Note that since the test items were excluded from training, we are simulating a scenario in which composition models must generate representations for nonce derived forms.

Cosine similarity between model-generated and corpus-extracted vectors were computed for all models, including the *stem* baseline (i.e., cosine between stem and derived form). The first row of Table 3 reports mean similarities. The *stem* method sets the level of performance relatively high, confirming its soundness. Indeed, the parameter-free *mult* model performs below the baseline.<sup>10</sup> As expected, *dilation* performs simi-

<sup>8</sup>More accurately, we relied on semi-manual CELEX information to identify derived forms. A further step towards a fully knowledge-free system would be to pre-process the corpus with an unsupervised morphological induction system to extract stem/derived pairs.

<sup>9</sup>The other models have thousands of weights to be estimated, so we cannot summarize the outcome of parameter estimation here.

<sup>10</sup>This result does not necessarily contradict those of

	<i>stem</i>	<i>mult</i>	<i>dil.</i>	<i>wadd</i>	<i>fulladd</i>	<i>lexfunc</i>
<b>All</b>	0.47	0.39	0.48	0.50	0.56	0.54
<b>HR</b>	0.52	0.43	0.53	0.55	0.61	0.58
<b>LR</b>	0.32	0.28	0.33	0.38	0.41	0.42

Table 3: Mean similarity of composed vectors to high-quality corpus-extracted derived-form vectors, for all as well as high- (HR) and low-relatedness (LR) test items

larly to the baseline, while *wadd* outperforms it, although the effect does not reach significance ( $p=.06$ ).<sup>11</sup> Both *fulladd* and *lexfunc* perform significantly better than *stem* ( $p < .001$ ). *Lexfunc* provides a flexible way to account for affixation, since it models it directly as a function mapping from and onto word vectors, without requiring a vector representation of bound affixes. The reason at the base of its good performance is thus quite straightforward. On the other hand, it is surprising that a simple representation of bound affixes (i.e., as vectors aggregating the contexts of words containing them) can work so well, at least when used in conjunction with the granular dimension-by-dimension weights assigned by the *fulladd* method. We hypothesize that these aggregated contexts, by providing information about the set of stems an affix combines with, capture the shared semantic features that the affix operates on.

When the meaning of the derived form is far from that of its stem, the *stem* baseline should no longer constitute a suitable surrogate of derived-form meaning. The LR cases (see Section 4.1 above) are thus crucial to understand how well composition methods capture not only stem meaning, but also affix-triggered semantics. The HR and LR rows of Table 3 present the results for the respective test subsets. As expected, the *stem* approach undergoes a strong drop when performance is measured on LR items. At the other extreme, *fulladd* and *lexfunc*, while also finding the LR cases more difficult, still clearly outperform the baseline ( $p < .001$ ), confirming that they capture the meaning of derived forms beyond what their stems contribute to it. The effect of *wadd*, again, approaches significance when compared to the baseline ( $p = .05$ ). Very encouragingly, both

Mitchell and Lapata and others who found *mult* to be highly competitive. Due to differences in co-occurrence weighting schemes (we use a logarithmically scaled measure, they do not), their multiplicative model is closer to our additive one.

<sup>11</sup>Significance assessed by means of Tukey Honestly Significant Difference tests (Abdi and Williams, 2010)

	<i>stem</i>	<i>mult</i>	<i>wadd</i>	<i>dil.</i>	<i>fulladd</i>	<i>lexfunc</i>
<b>-less</b>	0.22	0.23	0.30	0.24	0.38	0.44
<b>in-</b>	0.39	0.34	0.45	0.40	0.47	0.45
<b>un-</b>	0.33	0.33	0.41	0.34	0.44	0.46

Table 4: Mean similarity of composed vectors to high-quality corpus-extracted derived-form vectors with negative affixes

*fulladd* and *lexfunc* significantly outperform *stem* also in the HR subset ( $p < .001$ ). That is, the models provide better approximations of derived forms even when the stem itself should already be a good surrogate. The difference between the two models is not significant.

We noted in Section 4.1 that forms containing the “negative” affixes *-less*, *un-* and *in-* received on average low SDR scores, since negation impacts meaning more drastically than other operations. Table 4 reports the performance of the models on these affixes. Indeed, the *stem* baseline performs quite poorly, whereas *fulladd*, *lexfunc* and, to a lesser extent, *wadd* are quite effective in this condition as well, all performing greatly above the baseline. These results are intriguing in light of the fact that modeling negation is a challenging task for DSMs (Mohammad et al., 2013) as well as cDSMs (Preller and Sadrzadeh, 2011). To the extent that our best methods have captured the negating function of a prefix such as *in-*, they might be applied to tasks such as recognizing lexical opposites, or even simple forms of syntactic negation (modeling *inoperable* is just a short step away from modeling *not operable* compositionally).

## 6 Experiment 2: Comparing the quality of corpus-extracted and compositionally generated words

The first experiment simulated the scenario in which derived forms are not in our corpus, so that directly extracting their representation from it is not an option. The second experiment tests if compositionally-derived representations can be better than those extracted directly from the corpus when the latter is a possible strategy (i.e., the derived forms are attested in the source corpus). To this purpose, we focused on those 277 test items that were judged as low-quality (LQ, see Section 4.1), which are presumably more challenging to generate, and where the compositional route could be most useful.

We evaluated the derived forms generated by

	<i>corpus</i>	<i>stem</i>	<i>wadd</i>	<i>fulladd</i>	<i>lexfunc</i>
<b>All</b>	2.28	3.26	4.12	3.99	3.09
<b>HR</b>	2.29	3.56	4.48	4.31	3.31
<b>LR</b>	2.22	2.48	3.14	3.12	2.52

Table 5: Average quality ratings of derived vectors

Target	Model	Neighbors
florist	<i>wadd</i>	flora, fauna, ecosystem
	<i>fulladd</i>	flora, fauna, egologist
	<i>lexfunc</i>	ornithologist, naturalist, botanist
sparsity	<i>wadd</i>	sparse, sparsely, dense
	<i>fulladd</i>	sparse, sparseness, angularity
	<i>lexfunc</i>	fragility, angularity, smallness
inducement	<i>wadd</i>	induce, inhibit, inhibition
	<i>fulladd</i>	induce, inhibition, mediate
	<i>lexfunc</i>	impairment, cerebral, ocular
inoperable	<i>wadd</i>	operable, palliation, biopsy
	<i>fulladd</i>	operable, inoperative, ventilator
	<i>lexfunc</i>	inoperative, unavoidably, flaw
rename	<i>wadd</i>	name, later, namesake
	<i>fulladd</i>	name, namesake, later
	<i>lexfunc</i>	temporarily, reinstate, thereafter

Table 6: Examples of model-predicted neighbors for words with LQ corpus-extracted vectors

the models that performed best in the first experiment (*fulladd*, *lexfunc* and *wadd*), as well as the *stem* baseline, by means of another crowdsourcing study. We followed the same procedure used to assess the quality of corpus-extracted vectors, that is, we asked judges to rate the relatedness of the target forms to their NNs (we obtained on average 29 responses per form).

The first line of Table 5 reports the average quality (on a 7-point scale) of the representations of the derived forms as produced by the models and baseline, as well as of the corpus-harvested ones (*corpus* column). All compositional models produce representations that are of significantly higher quality ( $p < .001$ ) than the corpus-based ones. The effect is also evident in qualitative terms. Table 6 presents the NNs predicted by the three compositional methods for the same LQ test items whose corpus-based NNs are presented in Table 2. These results indicate that morpheme composition is an effective solution when the quality of corpus-extracted derived forms is low (and the previous experiment showed that, when their quality is high, composition can at least approximate corpus-based vectors).

With respect to Experiment 1, we obtain a different ranking of the models, with *lexfunc* being outperformed by both *wadd* and *fulladd* ( $p < .001$ ), that are statistically indistinguishable. The *wadd*

composition is dominated by the stem, and by looking at the examples in Table 6 we notice that both this model and *fulladd* tend to feature the stem as NN (100% of the cases for *wadd*, 73% for *fulladd* in the complete test set). The question thus arises as to whether the good performance of these composition techniques is simply due to the fact that they produce derived forms that are near their stems, with no added semantic value from the affix (a “stemploitation” strategy).

However, the stemploitation hypothesis is dispelled by the observation that both models significantly outperform the *stem* baseline ( $p < .001$ ), despite the fact that the latter, again, has good performance, significantly outperforming the corpus-derived vectors ( $p < .001$ ). Thus, we confirm that compositional models provide higher quality vectors that are capturing the meaning of derived forms beyond the information provided by the stem.

Indeed, if we focus on the third row of Table 5, reporting performance on low stem-derived relatedness (LR) items (annotated as described in Section 4.1), *fulladd* and *wadd* still significantly outperform the corpus representations ( $p < .001$ ), whereas the quality of the *stem* representations of LR items is not significantly different from that of the corpus-derived ones. Interestingly, *lexfunc* displays the smallest drop in performance when restricting evaluation to LR items; however, since it does not significantly outperform the LQ corpus representations, this is arguably due to a floor effect.

## 7 Conclusion and future work

We investigated to what extent cDSMs can generate effective meaning representations of complex words through morpheme composition. Several state-of-the-art composition models were adapted and evaluated on this novel task. Our results suggest that morpheme composition can indeed provide high-quality vectors for complex forms, improving both on vectors directly extracted from the corpus and on a stem-backoff strategy. This result is of practical importance for distributional semantics, as it paves the way to address one of the main causes of data sparseness, and it confirms the usefulness of the compositional approach in a new domain. Overall, *fulladd* emerged as the best performing model, with both *lexfunc* and the simple *wadd* approach constituting strong rivals. The ef-

fectiveness of the best models extended also to the challenging cases where the meaning of derived forms is far from that of the stem, including negative affixes.

The *fulladd* method requires a vector representation for bound morphemes. A first direction for future work will thus be to investigate which aspects of the meaning of bound morphemes are captured by our current simple-minded approach to populating their vectors, and to explore alternative ways to construct them, seeing if they further improve *fulladd* performance.

A natural extension of our research is to address morpheme composition and morphological induction jointly, trying to model the intuition that good candidate morphemes should have coherent semantic representations. Relatedly, in the current setting we generate complex forms from their parts. We want to investigate the inverse route, namely “de-composing” complex words to derive representations of their stems, especially for cases where the complex words are more frequent (e.g. *comfort/comfortable*).

We would also like to apply composition to inflectional morphology (that currently lies outside the scope of distributional semantics), to capture the nuances of meaning that, for example, distinguish singular and plural nouns (consider, e.g., the difference between the mass singular *tea* and the plural *teas*, which coerces the noun into a count interpretation (Katz and Zamparelli, 2012)).

Finally, in our current setup we focus on a single composition step, e.g., we derive the meaning of *inoperable* by composing the morphemes *in-* and *operable*. But *operable* is in turn composed of *operate* and *-able*. In the future, we will explore recursive morpheme composition, especially since we would like to apply these methods to more complex morphological systems (e.g., agglutinative languages) where multiple morphemes are the norm.

## 8 Acknowledgments

We thank Georgiana Dinu and Nghia The Pham for helping out with DISSECT-ion and the reviewers for helpful feedback. This research was supported by the ERC 2011 Starting Independent Research Grant n. 283554 (COMPOSES).

## References

- Hervé Abdi and Lynne Williams. 2010. Newman-Keuls and Tukey test. In Neil Salkind, Bruce Frey, and Donald Dougherty, editors, *Encyclopedia of Research Design*, pages 897–904. Sage, Thousand Oaks, CA.
- Harald Baayen, Richard Piepenbrock, and Leon Gulikers. 1995. The CELEX lexical database (release 2). CD-ROM, Linguistic Data Consortium, Philadelphia, PA.
- Harald Baayen. 2005. Morphological productivity. In Rajmund Piotrowski Reinhard Köhler, Gabriel Altmann, editor, *Quantitative Linguistics: An International Handbook*, pages 243–256. Mouton de Gruyter, Berlin, Germany.
- Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of EMNLP*, pages 1183–1193, Boston, MA.
- Marco Baroni. 2009. Distributions in text. In Anke Lüdeling and Merja Kytö, editors, *Corpus Linguistics: An International Handbook*, volume 2, pages 803–821. Mouton de Gruyter, Berlin, Germany.
- Laurie Bauer. 2001. *Morphological Productivity*. Cambridge University Press, Cambridge, UK.
- Kenneth Beesley and Lauri Karttunen. 2000. *Finite-State Morphology: Xerox Tools and Techniques*. Cambridge University Press, Cambridge, UK.
- Alan Black, Stephen Pulman, Graeme Ritchie, and Graham Russell. 1991. *Computational Morphology*. MIT Press, Cambridge, MA.
- Gemma Boleda, Marco Baroni, Louise McNally, and Nghia Pham. 2013. Intensionality was only alleged: On adjective-noun composition in distributional semantics. In *Proceedings of IWCS*, pages 35–46, Potsdam, Germany.
- John Bullinaria and Joseph Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39:510–526.
- Stephen Clark. 2012. Vector space models of lexical meaning. In Shalom Lappin and Chris Fox, editors, *Handbook of Contemporary Semantics, 2nd edition*. Blackwell, Malden, MA. In press.
- Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36:345–384.
- David Dowty. 1979. *Word Meaning and Montague Grammar*. Springer, New York.
- Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a Dirichlet process mixture model. In *Proceedings of EMNLP*, pages 616–627, Edinburgh, UK.
- Katrin Erk. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.
- Yoav Goldberg and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *Proceedings of ACL*, pages 371–379, Columbus, OH.
- John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 2(27):153–198.
- Sharon Goldwater and David McClosky. 2005. Improving statistical MT through morphological analysis. In *Proceedings of EMNLP*, pages 676–683, Vancouver, Canada.
- Sharon Goldwater. 2006. *Nonparametric Bayesian Models of Lexical Acquisition*. Ph.D. thesis, Brown University.
- Emiliano Guevara. 2009. Compositionality in distributional semantics: Derivational affixes. In *Proceedings of the Words in Action Workshop*, Pisa, Italy.
- Emiliano Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of GEMS*, pages 33–37, Uppsala, Sweden.
- Graham Katz and Roberto Zamparelli. 2012. Quantifying count/mass elasticity. In *Proceedings of WCFL*, pages 371–379, Tucson, AR.
- Victor Kuperman. 2009. Semantic transparency revisited. Presentation at the 6th International Morphological Processing Conference.
- Thomas Landauer and Susan Dumais. 1997. A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.
- Young-Suk Lee. 2004. Morphological analysis for statistical machine translation. In *Proceedings of HLT-NAACL*, pages 57–60, Boston, MA.
- Rochelle Lieber. 2004. *Morphology and Lexical Semantics*. Cambridge University Press, Cambridge, UK.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL*, pages 236–244, Columbus, OH.
- Jeff Mitchell and Mirella Lapata. 2009. Language models based on semantic composition. In *Proceedings of EMNLP*, pages 430–439, Singapore.

- Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.
- Saif Mohammad, Bonnie Dorr, Graeme Hirst, and Peter Turney. 2013. Computing lexical contrast. *Computational Linguistics*. In press.
- Jason Naradowsky and Sharon Goldwater. 2009. Improving morphology induction by learning spelling rules. In *Proceedings of IJCAI*, pages 11–17, Pasadena, CA.
- Ingo Plag. 1999. *Morphological Productivity: Structural Constraints in English Derivation*. Mouton de Gruyter, Berlin, Germany.
- Anne Preller and Mehrnoosh Sadrzadeh. 2011. Bell states and negative sentences in the distributed model of meaning. *Electr. Notes Theor. Comput. Sci.*, 270(2):141–153.
- Patrick Schone and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of the ConLL workshop on learning language in logic*, pages 67–72, Lisbon, Portugal.
- Richard Socher, Eric Huang, Jeffrey Pennin, Andrew Ng, and Christopher Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of NIPS*, pages 801–809, Granada, Spain.
- Richard Socher, Brody Huval, Christopher Manning, and Andrew Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP*, pages 1201–1211, Jeju Island, Korea.
- Richard Sproat. 1992. *Morphology and Computation*. MIT Press, Cambridge, MA.
- Peter Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Hsueh-Cheng Wang, Yi-Min Tien, Li-Chuan Hsu, and Marc Pomplun. 2012. Estimating semantic transparency of constituents of English compounds and two-character Chinese words using Latent Semantic Analysis. In *Proceedings of CogSci*, pages 2499–2504, Sapporo, Japan.
- Richard Wicentowski. 2004. Multilingual noise-robust supervised morphological analysis using the wordframe model. In *Proceedings of SIGPHON*, pages 70–77, Barcelona, Spain.
- David Yarowsky and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of ACL*, pages 207–216, Hong Kong.
- Fabio Zanzotto, Ioannis Korkontzelos, Francesca Falucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of COLING*, pages 1263–1271, Beijing, China.

# Unsupervised Consonant-Vowel Prediction over Hundreds of Languages

Young-Bum Kim and Benjamin Snyder

University of Wisconsin-Madison  
{ybkim,bsnyder}@cs.wisc.edu

## Abstract

In this paper, we present a solution to one aspect of the decipherment task: the prediction of consonants and vowels for an unknown language and alphabet. Adopting a classical Bayesian perspective, we perform posterior inference over hundreds of languages, leveraging knowledge of known languages and alphabets to uncover general linguistic patterns of typologically coherent language clusters. We achieve average accuracy in the unsupervised consonant/vowel prediction task of 99% across 503 languages. We further show that our methodology can be used to predict more fine-grained phonetic distinctions. On a three-way classification task between vowels, nasals, and non-nasal consonants, our model yields unsupervised accuracy of 89% across the same set of languages.

## 1 Introduction

Over the past centuries, dozens of lost languages have been deciphered through the painstaking work of scholars, often after decades of slow progress and dead ends. However, several important writing systems and languages remain undeciphered to this day.

In this paper, we present a successful solution to one aspect of the decipherment puzzle: automatically identifying basic phonetic properties of letters in an unknown alphabetic writing system. Our key idea is to use knowledge of the phonetic regularities encoded in known language vocabularies to automatically build a universal probabilistic model to successfully decode new languages.

Our approach adopts a classical Bayesian perspective. We assume that each language has an unobserved set of parameters explaining its

observed vocabulary. We further assume that each language-specific set of parameters was itself drawn from an unobserved common prior, shared across a cluster of typologically related languages. In turn, each cluster derives its parameters from a universal prior common to all language groups. This approach allows us to mix together data from languages with various levels of observations and perform joint posterior inference over unobserved variables of interest.

At the bottom layer (see Figure 1), our model assumes a language-specific data generating HMM over words in the language vocabulary. Each word is modeled as an emitted sequence of characters, depending on a corresponding Markov sequence of phonetic tags. Since individual letters are highly constrained in their range of phonetic values, we make the assumption of one-tag-per-observation-type (e.g. a single letter is constrained to be always a consonant or always a vowel across all words in a language).

Going one layer up, we posit that the language-specific HMM parameters are themselves drawn from informative, non-symmetric distributions representing a typologically coherent language grouping. By applying the model to a mix of languages with observed and unobserved phonetic sequences, the cluster-level distributions can be inferred and help guide prediction for unknown languages and alphabets.

We apply this approach to two small decipherment tasks:

1. predicting whether individual characters in an unknown alphabet and language represent vowels or consonants, and
2. predicting whether individual characters in an unknown alphabet and language represent vowels, nasals, or non-nasal consonants.

For both tasks, our approach yields considerable

success. We experiment with a data set consisting of vocabularies of 503 languages from around the world, written in a mix of Latin, Cyrillic, and Greek alphabets. In turn for each language, we consider it and its alphabet “unobserved” — we hide the graphic and phonetic properties of the symbols — while treating the vocabularies of the remaining languages as fully observed with phonetic tags on each of the letters.

On average, over these 503 leave-one-language-out scenarios, our model predicts consonant/vowel distinctions with 99% accuracy. In the more challenging task of vowel/nasal/non-nasal prediction, our model achieves average accuracy over 89%.

## 2 Related Work

The most direct precedent to the present work is a section in Knight et al. (2006) on universal phonetic decipherment. They build a trigram HMM with three hidden states, corresponding to consonants, vowels, and spaces. As in our model, individual characters are treated as the observed emissions of the hidden states. In contrast to the present work, they allow letters to be emitted by multiple states.

Their experiments show that the HMM trained with EM successfully clusters Spanish letters into consonants and vowels. They further design a more sophisticated finite-state model, based on linguistic universals regarding syllable structure and sonority. Experiments with the second model indicate that it can distinguish sonorous consonants (such as n, m, l, r) from non-sonorous consonants in Spanish. An advantage of the linguistically structured model is that its predictions do not require an additional mapping step from uninterpreted hidden states to linguistic categories, as they do with the HMM.

Our model and experiments can be viewed as complementary to the work of Knight et al., while also extending it to hundreds of languages. We use the simple HMM with EM as our baseline. In lieu of a linguistically designed model structure, we choose an empirical approach, allowing posterior inference over hundreds of known languages to guide the model’s decisions for the unknown script and language.

In this sense, our model bears some similarity to the decipherment model of Snyder et al. (2010), which used knowledge of a related language (Hebrew) in an elaborate Bayesian framework to de-

cipher the ancient language of Ugaritic. While the aim of the present work is more modest (discovering very basic phonetic properties of letters) it is also more widely applicable, as we don’t require detailed analysis of a known related language.

Other recent work has employed a similar perspective for tying learning across languages. Naseem et al. (2009) use a non-parametric Bayesian model over parallel text to jointly learn part-of-speech taggers across 8 languages, while Cohen and Smith (2009) develop a shared logistic normal prior to couple multilingual learning even in the absence of parallel text. In similar veins, Berg-Kirkpatrick and Klein (2010) develop hierarchically tied grammar priors over languages within the same family, and Bouchard-Côté et al. (2013) develop a probabilistic model of sound change using data from 637 Austronesian languages.

In our own previous work, we have developed the idea that *supervised* knowledge of some number of languages can help guide the unsupervised induction of linguistic structure, even in the absence of parallel text (Kim et al., 2011; Kim and Snyder, 2012)<sup>1</sup>. In the latter work we also tackled the problem of unsupervised phonemic prediction for unknown languages by using textual regularities of known languages. However, we assumed that the target language was written in a known (Latin) alphabet, greatly reducing the difficulty of the prediction task. In our present case, we assume no knowledge of any relationship between the writing system of the target language and known languages, other than that they are all alphabetic in nature.

Finally, we note some similarities of our model to some ideas proposed in other contexts. We make the assumption that each observation type (letter) occurs with only one hidden state (consonant or vowel). Similar constraints have been developed for part-of-speech tagging (Lee et al., 2010; Christodoulopoulos et al., 2011), and the power of type-based sampling has been demonstrated, even in the absence of explicit model constraints (Liang et al., 2010).

## 3 Model

Our generative Bayesian model over the observed vocabularies of hundreds of languages is

---

<sup>1</sup>We note that similar ideas were simultaneously proposed by other researchers (Cohen et al., 2011).

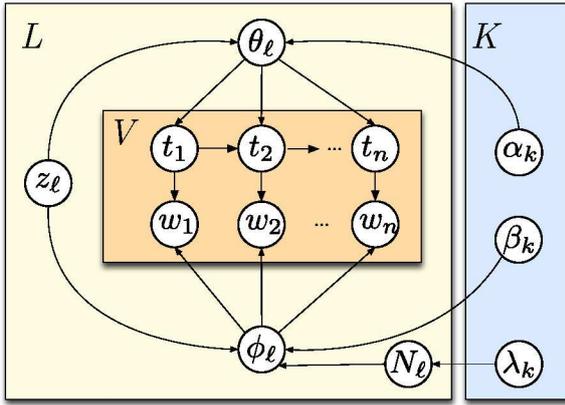


Figure 1: Graphical representation of our model. We have  $K$  language clusters,  $L$  languages, and  $V$  words in each language.

presented in Figure 1 and Algorithms 1, 2, and 3. We present a running commentary on the generative process from the bottom up, starting with Algorithm 3.

### 3.1 Data Generation

At the data generation stage (Algorithm 3), our model resembles an HMM. At each time step  $i$ , a tag  $t_i$  is selected according to a language-specific transition distribution  $\phi$ , indexed by the previous tag  $t_{i-1}$ . We note that in practice, we implemented a trigram version of the model,<sup>2</sup> but we present the bigram version here for notational clarity. We assume that our tagset includes phonetic categories of interest (such as consonant, vowel, nasal, etc) as well as a special tag to denote the boundaries between words.

An observation index  $j \in 1 \dots N_{\ell, t_i}$  is then drawn from the language-specific emission distribution  $\phi$ , indexed by the current tag  $t_i$ .  $N_{\ell, t_i}$  denotes the number of observation types associated with tag  $t_i$  in language  $\ell$ . Finally, we assume the existence of a deterministic function `orth` which maps each tag's observation indices to unique orthographic character symbols. This ensures that each observed character type corresponds to an observation index in exactly one tag category.

### 3.2 Language Generation

At the next stage up (Algorithm 2), we consider the generation of all language-specific parameters. This process begins by selecting a language cluster assignment  $z_\ell$  uniformly. The language cluster

<sup>2</sup>where the transition distribution is indexed by the previous *two* tags

---

#### Algorithm 1 Cluster Generation

---

```

for each cluster  $k \in 1 \dots K$  do
  for each tag  $t \in 1 \dots T$  do
    // emission Dirichlet parameter
     $\beta_{k,t} \sim \text{Unif}[0, 500]$ 

    // type-count Poisson parameter
     $\lambda_{k,t} \sim \text{Gamma}(g_1, g_2)$ 

    // transition Dirichlet parameters
    for each tag  $t'$  do
       $\alpha_{k,t,t'} \sim \text{Unif}[0, 500]$ 

```

---



---

#### Algorithm 2 Language Generation

---

```

for each language  $\ell$  do
  // draw cluster assignment
  cluster  $z_\ell \sim \text{Unif}[1 \dots K]$ 

  for each tag  $t$  do
    // generate tag type-count
     $N_{\ell,t} \sim \text{Poisson}(\lambda_{z_\ell,t})$ 

    // generate emission multinomial
     $\phi_{\ell,t,1} \dots \phi_{\ell,t,N_{\ell,t}} \sim \text{SymmDir}(\beta_{z_\ell,t})$ 

    // generate transition multinomial
     $\theta_{\ell,t,1} \dots \theta_{\ell,t,T} \sim \text{Dir}(\alpha_{z_\ell,t,1} \dots \alpha_{z_\ell,t,T})$ 

```

---



---

#### Algorithm 3 Data Generation

---

```

for each language  $\ell$  do
  for each position  $i$  do
    // transition to new tag token
     $t_i | t_{i-1} \sim \text{Cat}(\theta_{\ell,t_{i-1},1} \dots \theta_{\ell,t_{i-1},T})$ 

    // emit observation index token
     $j | t_i \sim \text{Cat}(\phi_{\ell,t_i,1} \dots \phi_{\ell,t_i,N_{\ell,t_i}})$ 

    // transcribe index token as character
     $w_i \leftarrow \text{orth}(\ell, j, t_i)$ 

```

---

provides priors over the HMM parameters. These priors include:

1. Poisson distributions over the number of observation types  $N_{\ell,t}$  associated with tag  $t$ ,
2. Dirichlet priors over transition distributions  $\theta$ , and
3. Dirichlet priors over emission distributions  $\phi$ .

For example, the cluster Poisson parameter over vowel observation types might be  $\lambda = 9$  (indicating 9 vowel letters on average for the cluster), while the parameter over consonant observation types might be  $\lambda = 20$  (indicating 20 consonant letters on average). These priors will be distinct for each language cluster and serve to characterize its general linguistic and typological properties.

We pause at this point to review the Dirichlet distribution in more detail. A  $k$ -dimensional Dirichlet with parameters  $\alpha_1 \dots \alpha_k$  defines a distribution over the  $k - 1$  simplex with the following density:

$$f(\theta_1 \dots \theta_k | \alpha_1 \dots \alpha_k) \propto \prod_i \theta_i^{\alpha_i - 1}$$

where  $\alpha_i > 0$ ,  $\theta_i > 0$ , and  $\sum_i \theta_i = 1$ . The Dirichlet serves as the *conjugate prior* for the Multinomial, meaning that the posterior  $\theta_1 \dots \theta_k | X_1 \dots X_n$  is again distributed as a Dirichlet (with updated parameters). It is instructive to reparameterize the Dirichlet with  $k + 1$  parameters:

$$f(\theta_1 \dots \theta_k | \alpha_0, \alpha'_1 \dots \alpha'_k) \propto \prod_i \theta_i^{\alpha_0 \alpha'_i - 1}$$

where  $\alpha_0 = \sum_i \alpha_i$ , and  $\alpha'_i = \alpha_i / \alpha_0$ . In this parameterization, we have  $\mathbb{E}[\theta_i] = \alpha'_i$ . In other words, the parameters  $\alpha'_i$  give the mean of the distribution, and  $\alpha_0$  gives the *precision* of the distribution. For large  $\alpha_0 \gg k$ , the distribution is highly peaked around the mean (conversely, when  $\alpha_0 \ll k$ , the mean lies in a valley).

Thus, the Dirichlet parameters of a language cluster characterize both the average HMMs of individual languages within the cluster, as well as how much we expect the HMMs to vary from the mean. In the case of emission distributions, we assume symmetric Dirichlet priors — i.e. one-parameter Dirichlets with densities given by  $f(\theta_1 \dots \theta_k | \beta) \propto \prod_i \theta_i^{\beta - 1}$ . This assumption is necessary, as we have no way to identify characters across languages in the decipherment scenario, and even the number of consonants and vowels (and thus multinomial/Dirichlet dimensions) can vary across the languages of a cluster. Thus, the mean of these Dirichlets will always be a uniform emission distribution. The single Dirichlet emission parameter per cluster will specify whether this mean is on a peak (large  $\beta$ ) or in a valley (small  $\beta$ ). In other words, it will control the expected *sparsity* of the resulting per-language emission multinomials.

In contrast, the transition Dirichlet parameters may be asymmetric, and thus very specific and informative. For example, one cluster may have the property that CCC consonant clusters are exceedingly rare across all its languages. This property would be expressed by a very small mean  $\alpha'_{\text{CCC}} \ll 1$  but large precision  $\alpha_0$ . Later we shall see examples of learned transition Dirichlet parameters.

### 3.3 Cluster Generation

The generation of the cluster parameters (Algorithm 1) defines the highest layer of priors for our model. As Dirichlets lack a standard conjugate prior, we simply use uniform priors over the interval  $[0, 500]$ . For the cluster Poisson parameters, we use conjugate Gamma distributions with vague priors.<sup>3</sup>

## 4 Inference

In this section we detail the inference procedure we followed to make predictions under our model. We run the procedure over data from 503 languages, assuming that all languages but one have observed character and tag sequences:  $w_1, w_2, \dots, t_1, t_2, \dots$ . Since each character type  $w$  is assumed to have a single tag category, this is equivalent to observing the character token sequence along with a character-type-to-tag mapping  $t_w$ . For the target language, we observe only character token sequence  $w_1, w_2, \dots$ .

We assume fixed and known parameter values only at the cluster generation level. Unobserved variables include (i) the cluster parameters  $\alpha, \beta, \lambda$ , (ii) the cluster assignments  $\mathbf{z}$ , (iii) the per-language HMM parameters  $\theta, \phi$  for all languages, and (iv) for the target language, the tag tokens  $t_1, t_2, \dots$  — or equivalently the character-type-to-tag mappings  $t_w$  — along with the observation type-counts  $N_t$ .

### 4.1 Monte Carlo Approximation

Our goal in inference is to predict the most likely tag  $t_{w,\ell}$  for each character type  $w$  in our target language  $\ell$  according to the posterior:

$$f(t_{w,\ell} | \mathbf{w}, \mathbf{t}_{-\ell}) = \int f(\mathbf{t}_\ell, \mathbf{z}, \alpha, \beta | \mathbf{w}, \mathbf{t}_{-\ell}) d\Theta \quad (1)$$

<sup>3</sup>(1,19) for consonants, (1,10) for vowels, (0.2, 15) for nasals, and (1,16) for non-nasal consonants.

where  $\Theta = (\mathbf{t}_{-w,\ell}, \mathbf{z}, \alpha, \beta)$ ,  $\mathbf{w}$  are the observed character sequences for all languages,  $\mathbf{t}_{-\ell}$  are the character-to-tag mappings for the observed languages,  $\mathbf{z}$  are the language-to-cluster assignments, and  $\alpha$  and  $\beta$  are all the cluster-level transition and emission Dirichlet parameters.

Sampling values  $(\mathbf{t}_\ell, \mathbf{z}, \alpha, \beta)_{n=1}^N$  from the integrand in Equation 1 allows us to perform the standard Monte Carlo approximation:

$$f(t_{w,\ell} = t \mid \mathbf{w}, \mathbf{t}_{-\ell}) \approx N^{-1} \sum_{n=1}^N \mathbb{I}(t_{w,\ell} = t \text{ in sample } n) \quad (2)$$

To maximize the Monte Carlo posterior, we simply take the most commonly sampled tag value for character type  $w$  in language  $\ell$ . Note that we leave out the language-level HMM parameters  $(\theta, \phi)$  as well as the cluster-level Poisson parameters  $\lambda$  from Equation 1 (and thus our sample space), as we can analytically integrate them out in our sampling equations.

## 4.2 Gibbs Sampling

To sample values  $(\mathbf{t}_\ell, \mathbf{z}, \alpha, \beta)$  from their posterior (the integrand of Equation 1), we use Gibbs sampling, a Monte Carlo technique that constructs a Markov chain over a high-dimensional sample space by iteratively sampling each variable conditioned on the currently drawn sample values for the others, starting from a random initialization. The Markov chain converges to an equilibrium distribution which is in fact the desired joint density (Geman and Geman, 1984). We now sketch the sampling equations for each of our sampled variables.

### Sampling $t_{w,\ell}$

To sample the tag assignment to character  $w$  in language  $\ell$ , we need to compute:

$$f(t_{w,\ell} \mid \mathbf{w}, \mathbf{t}_{-w,\ell}, \mathbf{t}_{-\ell}, \mathbf{z}, \alpha, \beta) \quad (3)$$

$$\propto f(\mathbf{w}_\ell, \mathbf{t}_\ell, N_\ell \mid \alpha_k, \beta_k, \mathbf{N}_{k-\ell}) \quad (4)$$

where  $N_\ell$  are the types-per-tag counts implied by the mapping  $\mathbf{t}_\ell$ ,  $k$  is the current cluster assignment for the target language ( $z_\ell = k$ ),  $\alpha_k$  and  $\beta_k$  are the cluster parameters, and  $\mathbf{N}_{k-\ell}$  are the types-per-tag counts for all languages currently assigned to the cluster, *other* than language  $\ell$ .

Applying the chain rule along with our model's conditional independence structure, we can further

re-write Equation 4 as a product of three terms:

$$f(N_\ell \mid \mathbf{N}_{k-\ell}) \quad (5)$$

$$f(t_1, t_2, \dots \mid \alpha_k) \quad (6)$$

$$f(w_1, w_2, \dots \mid N_\ell, t_1, t_2, \dots, \beta_k) \quad (7)$$

The first term is the posterior predictive distribution for the Poisson-Gamma compound distribution and is easy to derive. The second term is the tag transition predictive distribution given Dirichlet hyperparameters, yielding a familiar Polya urn scheme form. Removing terms that don't depend on the tag assignment  $t_{\ell,w}$  gives us:

$$\frac{\prod_{t,t'} (\alpha_{k,t,t'} + n(t,t'))^{[n'(t,t')]} \prod_t (\sum_{t'} \alpha_{k,t,t'} + n(t))^{[n'(t)]}}$$

where  $n(t)$  and  $n(t,t')$  are, respectively, unigram and bigram tag counts *excluding* those containing character  $w$ . Conversely,  $n'(t)$  and  $n'(t,t')$  are, respectively, unigram and bigram tag counts *including* those containing character  $w$ . The notation  $a^{[n]}$  denotes the ascending factorial:  $a(a+1) \cdots (a+n-1)$ . Finally, we tackle the third term, Equation 7, corresponding to the predictive distribution of emission observations given Dirichlet hyperparameters. Again, removing constant terms gives us:

$$\frac{\beta_{k,t}^{[n(w)]}}{\prod_{t'} N_{\ell,t'} \beta_{k,t'}^{[n'(t')]}}$$

where  $n(w)$  is the unigram count of character  $w$ , and  $n'(t')$  is the unigram count of tag  $t'$ , over all characters tokens (including  $w$ ).

### Sampling $\alpha_{k,t,t'}$

To sample the Dirichlet hyperparameter for cluster  $k$  and transition  $t \rightarrow t'$ , we need to compute:

$$\begin{aligned} f(\alpha_{k,t,t'} \mid \mathbf{t}, \mathbf{z}) \\ \propto f(\mathbf{t}, \mathbf{z} \mid \alpha_{z,t,t'}) \\ = f(\mathbf{t}_k \mid \alpha_{z,t,t'}) \end{aligned}$$

where  $\mathbf{t}_k$  are the tag sequences for all languages currently assigned to cluster  $k$ . This term is a predictive distribution of the multinomial-Dirichlet compound when the observations are grouped into *multiple* multinomials all with the same prior. Rather than inefficiently computing a product of Polya urn schemes (with many repeated ascending

factorials with the same base), we group common terms together and calculate:

$$\frac{\prod_{j=1}^n (\alpha_{k,t,t'} + k)^{n(j,k,t,t')}}{\prod_{j=1}^n (\sum_{t''} \alpha_{k,t,t''} + k)^{n(j,k,t)}}$$

where  $n(j, k, t)$  and  $n(j, k, t, t')$  are the numbers of languages currently assigned to cluster  $k$  which have *more than*  $j$  occurrences of unigram ( $t$ ) and bigram ( $t, t'$ ), respectively.

This gives us an efficient way to compute unnormalized posterior densities for  $\alpha$ . However, we need to sample from these distributions, not just compute them. To do so, we turn to slice sampling (Neal, 2003), a simple yet effective auxiliary variable scheme for sampling values from unnormalized but otherwise computable densities.

The key idea is to supplement the variable  $x$ , distributed according to unnormalized density  $\tilde{p}(x)$ , with a second variable  $u$  with joint density defined as  $p(x, u) \propto \mathbb{I}(u < \tilde{p}(x))$ . It is easy to see that  $\tilde{p}(x) \propto \int p(x, u) du$ . We then iteratively sample  $u|x$  and  $x|u$ , both of which are distributed uniformly across appropriately bounded intervals. Our implementation follows the pseudocode given in Mackay (2003).

### Sampling $\beta_{k,t}$

To sample the Dirichlet hyperparameter for cluster  $k$  and tag  $t$  we need to compute:

$$\begin{aligned} f(\beta_{k,t} | \mathbf{t}, \mathbf{w}, \mathbf{z}, \mathbf{N}) \\ \propto f(\mathbf{w} | \mathbf{t}, \mathbf{z}, \beta_{k,t}, \mathbf{N}) \\ \propto f(\mathbf{w}_k | \mathbf{t}_k, \beta_{k,t}, \mathbf{N}_k) \end{aligned}$$

where, as before,  $\mathbf{t}_k$  are the tag sequences for languages assigned to cluster  $k$ ,  $\mathbf{N}_k$  are the tag observation type-counts for languages assigned to the cluster, and likewise  $\mathbf{w}_k$  are the character sequences of all languages in the cluster. Again, we have the predictive distribution of the multinomial-Dirichlet compound with multiple grouped observations. We can apply the same trick as above to group terms in the ascending factorials for efficient computation. As before, we use slice sampling for obtaining samples.

### Sampling $z_\ell$

Finally, we consider sampling the cluster assignment  $z_\ell$  for each language  $\ell$ . We calculate:

$$\begin{aligned} f(z_\ell = k | \mathbf{w}, \mathbf{t}, \mathbf{N}, \mathbf{z}_{-\ell}, \alpha, \beta) \\ \propto f(\mathbf{w}_\ell, \mathbf{t}_\ell, N_\ell | \alpha_k, \beta_k, \mathbf{N}_{k-\ell}) \\ = f(N_\ell | \mathbf{N}_{k-\ell}) f(\mathbf{t}_\ell | \alpha_k) f(\mathbf{w}_\ell | \mathbf{t}_\ell, N_\ell, \beta_k) \end{aligned}$$

The three terms correspond to (1) a standard predictive distributions for the Poisson-gamma compound and (2) the standard predictive distributions for the transition and emission multinomial-Dirichlet compounds.

## 5 Experiments

To test our model, we apply it to a corpus of 503 languages for two decipherment tasks. In both cases, we will assume no knowledge of our target language or its writing system, other than that it is alphabetic in nature. At the same time, we will assume basic phonetic knowledge of the writing systems of the other 502 languages. For our first task, we will predict whether each character type is a consonant or a vowel. In the second task, we further subdivide the consonants into two major categories: the nasal consonants, and the non-nasal consonants. Nasal consonants are known to be perceptually very salient and are unique in being high frequency consonants in all known languages.

### 5.1 Data

Our data is drawn from online electronic translations of the Bible (<http://www.bible.is>, <http://www.crosswire.org/index.jsp>, and <http://www.biblegateway.com>). We have identified translations covering 503 distinct languages employing alphabetic writing systems. Most of these languages (476) use variants of the Latin alphabet, a few (26) use Cyrillic, and one uses the Greek alphabet. As Table 1 indicates, the languages cover a very diverse set of families and geographic regions, with Niger-Congo languages being the largest represented family.<sup>4</sup> Of these languages, 30 are either language isolates, or sole members of their language family in our data set.

For our experiments, we extracted unique word types occurring at least 5 times from the downloaded Bible texts. We manually identified vowel, nasal, and non-nasal character types. Since the letter ‘‘y’’ can frequently represent both a consonant and vowel, we exclude it from our evaluation. On average, the resulting vocabularies contain 2,388 unique words, with 19 consonant characters, two 2 nasal characters, and 9 vowels. We include the data as part of the paper.

<sup>4</sup>In fact, the Niger-Congo grouping is often considered the largest language family in the world in terms of distinct member languages.

Language Family	#lang
Niger-Congo	114
Austronesian	67
Oto-Manguean	41
Indo-European	39
Mayan	34
Quechuan	17
Afro-Asiatic	17
Uto-Aztecan	16
Altaic	16
Trans-New Guinea	15
Nilo-Saharan	14
Sino-Tibetan	13
Tucanoan	9
Creole	8
Chibchan	6
Maipurean	5
Tupian	5
Nakh-Daghestanian	4
Uralic	4
Cariban	4
Totonacan	4
Mixe-Zoque	3
Jivaroan	3
Choco	3
Guajiboan	2
Huavean	2
Austro-Asiatic	2
Witotoan	2
Jean	2
Paezan	2
<i>Other</i>	30

Table 1: Language families in our data set. The *Other* category includes 9 language isolates and 21 language family singletons.

## 5.2 Baselines and Model Variants

As our baseline, we consider the trigram HMM model of Knight et al. (2006), trained with EM. In all experiments, we run 10 random restarts of EM, and pick the prediction with highest likelihood. We map the induced tags to the gold-standard tag categories (1-1 mapping) in the way that maximizes accuracy.

We then consider three variants of our model. The simplest version, SYMM, disregards all information from other languages, using simple symmetric hyperparameters on the transition and emission Dirichlet priors (all hyperparameters set to 1). This allows us to assess the performance of

	Model	Cons vs Vowel	C vs V vs N
All	EM	93.37	74.59
	SYMM	95.99	80.72
	MERGE	97.14	86.13
	CLUST	<b>98.85</b>	<b>89.37</b>
Isolates	EM	94.50	74.53
	SYMM	96.18	78.13
	MERGE	97.66	86.47
	CLUST	<b>98.55</b>	<b>89.07</b>
Non-Latin	EM	92.93	78.26
	SYMM	95.90	79.04
	MERGE	96.06	83.78
	CLUST	<b>97.03</b>	<b>85.79</b>

Table 2: Average accuracy for EM baseline and model variants across 503 languages. **First panel:** results on all languages. **Second panel:** results for 30 isolate and singleton languages. **Third panel:** results for 27 non-Latin alphabet languages (Cyrillic and Greek). Standard Deviations across languages are about 2%.

our Gibbs sampling inference method for the type-based HMM, even in the absence of multilingual priors.

We next consider a variant of our model, MERGE, that assumes that *all* languages reside in a single cluster. This allows knowledge from the other languages to affect our tag posteriors in a generic, language-neutral way.

Finally, we consider the full version of our model, CLUST, with 20 language clusters. By allowing for the division of languages into smaller groupings, we hope to learn more specific parameters tailored for typologically coherent clusters of languages.

## 6 Results

The results of our experiments are shown in Table 2. In all cases, we report token-level accuracy (i.e. frequent characters count more than infrequent characters), and results are macro-averaged over the 503 languages. Variance across languages is quite low: the standard deviations are about 2 percentage points.

For the consonant vs. vowel prediction task, all tested models perform well. Our baseline, the EM-based HMM, achieves 93.4% accuracy. Simply using our Gibbs sampler with symmetric priors boosts the performance up to 96%. Performance

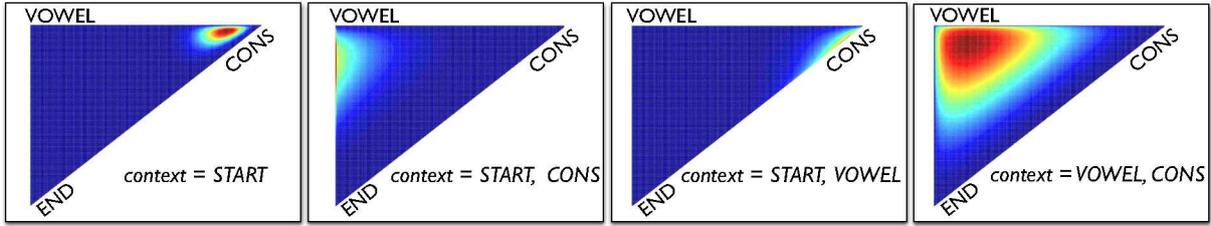


Figure 2: Inferred transition Dirichlet distributions for trigram MERGE model. Heat plots indicate Dirichlet densities over the 2-simplex.

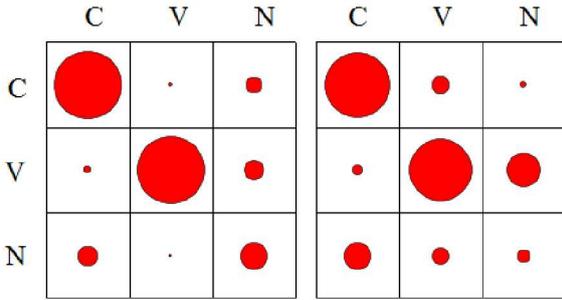


Figure 3: Confusion matrix for CLUST (left) and EM (right). Rows show true values, columns show predicted values. Size of blobs are proportional to counts.

increases again when we condition on other languages (MERGE), and we observe nearly 99% accuracy when allowing languages to cluster.

In the three-way nasal vs. non-nasal consonant vs. vowel prediction task, EM does not fare particularly well, only achieving 75% accuracy. As before, we see increasing performance gains for our model variants, culminating in almost 90% accuracy when the language clustering is used. The relatively weak performance of EM in this case should not be surprising: there is no *a priori* reason to expect any particular three-way classification to be the most salient clustering of letters from the perspective of EM. In contrast, our empirical multilingual approach allows the language-specific tag predictions to be guided by whatever values are set for the other, observed, languages.

We note that although a post-hoc mapping from inferred tags to true tags is necessary for both EM and SYMM, this is not the case for the final two variants of our model. Both MERGE and CLUST break symmetries over tags by way of the asymmetric posterior over transition Dirichlet parameters. Thus the reported accuracies are obtained without the need for any additional tag mappings.

Figure 2 further breaks down results for languages without any other related language in our collection. These include 9 language isolates and 21 singleton languages acting as sole representatives of their families. In addition, we show results for the 27 languages which employ non-Latin alphabets (26 Cyrillic and one Greek). Both of these scenarios are likely to occur in cases of lost language decipherment. We see similar results and trends, with somewhat lower performance in both cases.

## 7 Analysis

To further compare our model to the EM baseline, we show confusion matrices for the three-way classification task in Figure 3. We can immediately see that EM had considerable difficulty making nasal predictions. Most true nasals (third row) are assigned to the regular consonant category, and apparently EM mostly used the additional tag as a way to further subcategorize vowels. In contrast, our model does fairly well with nasals: most actual nasals are assigned to the nasal category (third row), while the plurality of nasal predictions are indeed true nasals (third column).

Next we examine the transition Dirichlet hyperparameters learned by our model. For the MERGE model, we infer a posterior over parameters shared by all 503 languages in our data set. Figure 2 shows MAP estimates of four of the Dirichlets governing transition probabilities from various contexts. As we can see, the learned hyperparameters yield highly asymmetric priors over transition distributions. Most languages like to start words with consonants, and after an initial consonant or vowel prefer to switch to the opposite category. In contrast, after a vowel-consonant sequence, languages can vary significantly in terms of the category favored next.

Figure 4 shows MAP transition Dirichlet hyperparameters of the CLUST model, when trained

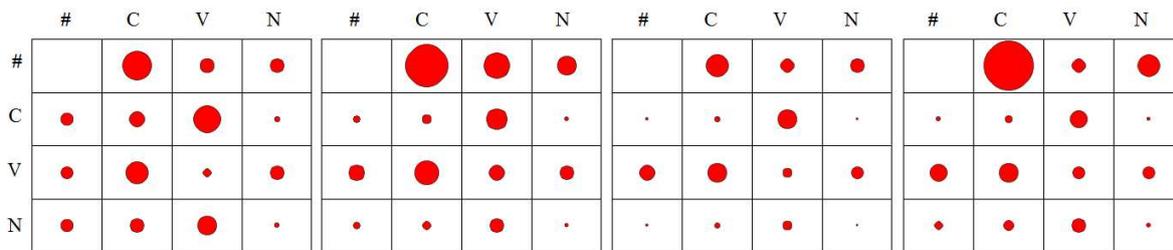


Figure 4: Inferred Dirichlet transition hyperparameters for bigram CLUST on three-way classification task with four latent clusters. Row gives starting state, column gives target state. Size of red blobs are proportional to magnitude of corresponding hyperparameters.

Language Family	Portion	#langs	Ent.
Indo-European	0.38	26	2.26
	0.24	41	3.19
	0.21	38	3.77
Quechuan	0.89	18	0.61
Mayan	0.64	33	1.70
Oto-Manguean	0.55	31	1.99
Maipurean	0.25	8	2.75
Tucanoan	0.2	45	3.98
Uto-Aztecan	0.4	25	2.85
Altaic	0.44	27	2.76
Niger-Congo	1	2	0.00
	0.78	23	1.26
	0.74	27	1.05
	0.68	22	1.22
	0.67	33	1.62
	0.5	18	2.21
Austronesian	0.24	25	3.27
	0.91	22	0.53
	0.71	21	1.51
	0.24	17	3.06

Table 3: Plurality language families across 20 clusters. The columns indicate portion of languages in the plurality family, number of languages, and entropy over families.

with a bigram HMM with four language clusters. Examining just the first row, we see that the languages are partially grouped by their preference for the initial tag of words. All clusters favor languages which prefer initial consonants, though this preference is most weakly expressed in cluster 3. In contrast, both clusters 2 and 4 have very dominant tendencies towards consonant-initial languages, but differ in the relative weight given to languages preferring either vowels or nasals initially.

Finally, we examine the relationship between the induced clusters and language families in Table 3, for the trigram consonant vs. vowel CLUST model with 20 clusters. We see that for about half the clusters, there is a majority language family, most often Niger-Congo. We also observe distinctive clusters devoted to Austronesian and Quechuan languages. The largest two clusters are rather indistinct, without any single language family achieving more than 24% of the total.

## 8 Conclusion

In this paper, we presented a successful solution to one aspect of the decipherment task: the prediction of consonants and vowels for an unknown language and alphabet. Adopting a classical Bayesian perspective, we develop a model that performs posterior inference over hundreds of languages, leveraging knowledge of known languages to uncover general linguistic patterns of typologically coherent language clusters. Using this model, we automatically distinguish between consonant and vowel characters with nearly 99% accuracy across 503 languages. We further experimented on a three-way classification task involving nasal characters, achieving nearly 90% accuracy.

Future work will take us in several new directions: first, we would like to move beyond the assumption of an alphabetic writing system so that we can apply our method to undeciphered syllabic scripts such as Linear A. We would also like to extend our methods to achieve finer-grained resolution of phonetic properties beyond nasals, consonants, and vowels.

## Acknowledgments

The authors thank the reviewers and acknowledge support by the NSF (grant IIS-1116676) and a research gift from Google. Any opinions, findings, or conclusions are those of the authors, and do not necessarily reflect the views of the NSF.

## References

- Taylor Berg-Kirkpatrick and Dan Klein. 2010. Phylogenetic grammar induction. In *Proceedings of the ACL*, pages 1288–1297. Association for Computational Linguistics.
- Alexandre Bouchard-Côté, David Hall, Thomas L Griffiths, and Dan Klein. 2013. Automated reconstruction of ancient languages using probabilistic models of sound change. *Proceedings of the National Academy of Sciences*, 110(11):4224–4229.
- Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2011. A Bayesian mixture model for part-of-speech induction using multiple features. In *Proceedings of EMNLP*, pages 638–647. Association for Computational Linguistics.
- Shay B Cohen and Noah A Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of NAACL*, pages 74–82. Association for Computational Linguistics.
- Shay B Cohen, Dipanjan Das, and Noah A Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of EMNLP*, pages 50–61. Association for Computational Linguistics.
- Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–741.
- Young-Bum Kim and Benjamin Snyder. 2012. Universal grapheme-to-phoneme prediction over latin alphabets. In *Proceedings of EMNLP*, pages 332–343, Jeju Island, South Korea, July. Association for Computational Linguistics.
- Young-Bum Kim, João V Graça, and Benjamin Snyder. 2011. Universal morphological analysis using structured nearest neighbor prediction. In *Proceedings of EMNLP*, pages 322–332. Association for Computational Linguistics.
- Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proceedings of COLING/ACL*, pages 499–506. Association for Computational Linguistics.
- Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2010. Simple type-level unsupervised POS tagging. In *Proceedings of EMNLP*, pages 853–861. Association for Computational Linguistics.
- Percy Liang, Michael I Jordan, and Dan Klein. 2010. Type-based MCMC. In *Proceedings of NAACL*, pages 573–581. Association for Computational Linguistics.
- David JC MacKay. 2003. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press.
- Tahira Naseem, Benjamin Snyder, Jacob Eisenstein, and Regina Barzilay. 2009. Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research*, 36(1):341–385.
- Radford M Neal. 2003. Slice sampling. *Annals of statistics*, 31:705–741.
- Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *Proceedings of the ACL*, pages 1048–1057. Association for Computational Linguistics.

# Improving Text Simplification Language Modeling Using Unsimplified Text Data

David Kauchak

Middlebury College

Middlebury, VT 05753

dkauchak@middlebury.edu

## Abstract

In this paper we examine language modeling for text simplification. Unlike some text-to-text translation tasks, text simplification is a monolingual translation task allowing for text in both the input and output domain to be used for training the language model. We explore the relationship between normal English and simplified English and compare language models trained on varying amounts of text from each. We evaluate the models intrinsically with perplexity and extrinsically on the lexical simplification task from SemEval 2012. We find that a combined model using both simplified and normal English data achieves a 23% improvement in perplexity and a 24% improvement on the lexical simplification task over a model trained only on simple data. Post-hoc analysis shows that the additional unsimplified data provides better coverage for unseen and rare  $n$ -grams.

## 1 Introduction

An important component of many text-to-text translation systems is the language model which predicts the likelihood of a text sequence being produced in the output language. In some problem domains, such as machine translation, the translation is between two distinct languages and the language model can only be trained on data in the output language. However, some problem domains (e.g. text compression, text simplification and summarization) can be viewed as monolingual translation tasks, translating between text variations within a single language. In these monolingual problems, text *could* be used from both the input and output domain to train a language model. In this paper, we investigate this possibility for text

simplification where both simplified English text and normal English text are available for training a simple English language model.

Table 1 shows the  $n$ -gram overlap proportions in a sentence aligned data set of 137K sentence pairs from aligning Simple English Wikipedia and English Wikipedia articles (Coster and Kauchak, 2011a).<sup>1</sup> The data highlights two conflicting views: does the benefit of additional data outweigh the problem of the source of the data? Throughout the rest of this paper we refer to sentences/articles/text from English Wikipedia as *normal* and sentences/articles/text from Simple English Wikipedia as *simple*.

On the one hand, there is a strong correspondence between the simple and normal data. At the word level 96% of the simple words are found in the normal corpus and even for  $n$ -grams as large as 5, more than half of the  $n$ -grams can be found in the normal text. In addition, the normal text does represent English text and contains many  $n$ -grams not seen in the simple corpus. This extra information may help with data sparsity, providing better estimates for rare and unseen  $n$ -grams.

On the other hand, there is still only modest overlap between the sentences for longer  $n$ -grams, particularly given that the corpus is sentence-aligned and that 27% of the sentence pairs in this aligned data set are identical. If the word distributions were very similar between simple and normal text, then the overlap proportions between the two languages would be similar regardless of which direction the comparison is made. Instead, we see that the normal text has more varied language and contains more  $n$ -grams. Previous research has also shown other differences between simple and normal data sources that could impact language model performance including average number of syllables, reading

<sup>1</sup><http://www.cs.middlebury.edu/~dkauchak/simplification>

<i>n</i> -gram size:	1	2	3	4	5
simple in normal	0.96	0.80	0.68	0.61	0.55
normal in simple	0.87	0.68	0.58	0.51	0.46

Table 1: The proportion of *n*-grams that overlap in a corpus of 137K sentence-aligned pairs from Simple English Wikipedia and English Wikipedia.

complexity, and grammatical complexity (Napoles and Dredze, 2010; Zhu et al., 2010; Coster and Kauchak, 2011b). In addition, for some monolingual translation domains, it has been argued that it is not appropriate to train a language model using data from the input domain (Turner and Charniak, 2005).

Although this question arises in other monolingual translation domains, text simplification represents an ideal problem area for analysis. First, simplified text data is available in reasonable quantities. Simple English Wikipedia contains more than 60K articles written in simplified English. This is not the case for all monolingual translation tasks (Knight and Marcu, 2002; Cohn and Lapata, 2009). Second, the quantity of simple text data available is still limited. After pre-processing, the 60K articles represents less than half a million sentences which is orders of magnitude smaller than the amount of normal English data available (for example the English Gigaword corpus (David Graff, 2003)). Finally, many recent text simplification systems have utilized language models trained only on simplified data (Zhu et al., 2010; Woodsend and Lapata, 2011; Coster and Kauchak, 2011a; Wubben et al., 2012); improvements in simple language modeling could translate into improvements for these systems.

## 2 Related Work

If we view the normal data as out-of-domain data, then the problem of combining simple and normal data is similar to the language model domain adaption problem (Suzuki and Gao, 2005), in particular *cross-domain adaptation* (Bellegarda, 2004) where a domain-specific model is improved by incorporating additional general data. Adaptation techniques have been shown to improve language modeling performance based on perplexity (Rosenfeld, 1996) and in application areas such as speech transcription (Bacchiani and Roark, 2003) and machine translation (Zhao et al., 2004), though no previous research has examined the lan-

guage model domain adaptation problem for text simplification. Pan and Yang (2010) provide a survey on the related problem of domain adaptation for machine learning (also referred to as “transfer learning”), which utilizes similar techniques. In this paper, we explore some basic adaptation techniques, however this paper is not a comparison of domain adaptation techniques for language modeling. Our goal is more general: to examine the relationship between simple and normal data and determine whether normal data is helpful. Previous domain adaptation research is complementary to our experiments and could be explored in the future for additional performance improvements.

Simple language models play a role in a variety of text simplification applications. Many recent statistical simplification techniques build upon models from machine translation and utilize a simple language model during simplification/decoding both in English (Zhu et al., 2010; Woodsend and Lapata, 2011; Coster and Kauchak, 2011a; Wubben et al., 2012) and in other languages (Specia, 2010). Simple English language models have also been used as predictive features in other simplification sub-problems such as lexical simplification (Specia et al., 2012) and predicting text simplicity (Eickhoff et al., 2010).

Due to data scarcity, little research has been done on language modeling in other monolingual translation domains. For text compression, most systems are trained on uncompressed data since the largest text compression data sets contain only a few thousand sentences (Knight and Marcu, 2002; Galley and McKeown, 2007; Cohn and Lapata, 2009; Nomoto, 2009). Similarly for summarization, systems that have employed language models trained only on unsummarized text (Banko et al., 2000; Daume and Marcu, 2002).

## 3 Corpus

We collected a data set from English Wikipedia and Simple English Wikipedia with the former representing normal English and the latter simple English. Simple English Wikipedia has been previously used for many text simplification approaches (Zhu et al., 2010; Yatskar et al., 2010; Biran et al., 2011; Coster and Kauchak, 2011a; Woodsend and Lapata, 2011; Wubben et al., 2012) and has been shown to be simpler than normal English Wikipedia by both automatic measures and human perception (Coster and Kauchak, 2011b;

	simple	normal
sentences	385K	2540K
words	7.15M	64.7M
vocab size	78K	307K

Table 2: Summary counts for the simple-normal article aligned data set consisting of 60K article pairs.

Woodsend and Lapata, 2011). We downloaded **all** articles from Simple English Wikipedia then removed stubs, navigation pages and any article that consisted of a single sentence, resulting in 60K simple articles.

To partially normalize for content and source differences we generated a document aligned corpus for our experiments. We extracted the corresponding 60K normal articles from English Wikipedia based on the article title to represent the normal data. We held out 2K article pairs for use as a testing set in our experiments. The extracted data set is available for download online.<sup>2</sup>

Table 2 shows count statistics for the collected data set. Although the simple and normal data contain the same number of articles, because normal articles tend to be longer and contain more content, the normal side is an order of magnitude larger.

## 4 Language Model Evaluation: Perplexity

To analyze the impact of data source on simple English language modeling, we trained language models on varying amounts of simple data, normal data, and a combination of the two. For our first task, we evaluated these language models using perplexity based on how well they modeled the simple side of the held-out data.

### 4.1 Experimental Setup

We used trigram language models with interpolated Kneser-Kney discounting trained using the SRI language modeling toolkit (Stolcke, 2002). To ensure comparability, all models were closed vocabulary with the same vocabulary set based on the words that occurred in the simple side of the training corpus, though similar results were seen for other vocabulary choices. We generated different models by varying the size and type of training

<sup>2</sup><http://www.cs.middlebury.edu/~dkauchak/simplification>

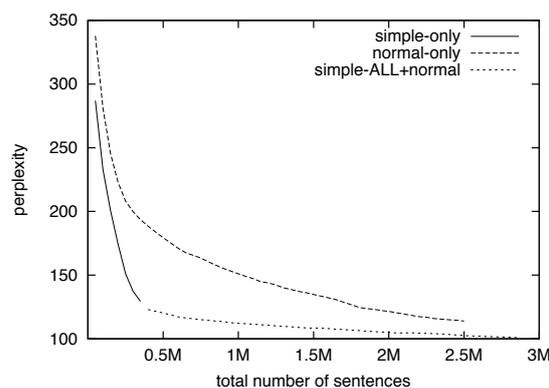


Figure 1: Language model perplexities on the held-out test data for models trained on increasing amounts of data.

data:

- **simple-only**: simple sentences only
- **normal-only**: normal sentences only
- **simple- $X$ +normal**:  $X$  simple sentences combined with a varying number of normal sentences

To evaluate the language models we calculated the model *perplexity* (Chen et al., 1998) on the simple side of the held-out data. The test set consisted of 2K simple English articles with 7,799 simple sentences and 179K words. Perplexity measures how likely a model finds a test set, with lower scores indicating better performance.

### 4.2 Perplexity Results

Figure 1 shows the language model perplexities for the three types of models for increasing amounts of training data. As expected, when trained on the same amount of data, the language models trained on simple data perform significantly better than language models trained on normal data. In addition, as we increase the amount of data, the simple-only model improves more than the normal-only model.

However, the results also show that the normal data does have some benefit. The perplexity for the *simple-ALL+normal* model, which starts with *all* available simple data, continues to improve as normal data is added resulting in a 23% improvement over the model trained with only simple data (from a perplexity of 129 down to 100). Even by itself the normal data does have value. The normal-only model achieves a slightly better perplexity than the simple-only model, though only by utilizing an order of magnitude more data.

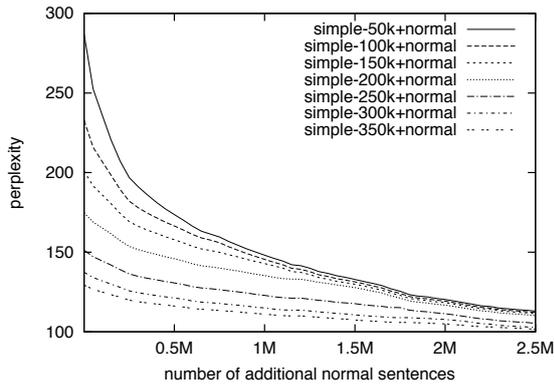


Figure 2: Language model perplexities for combined simple-normal models. Each line represents a model trained on a different amount of simple data as normal data is added.

To better understand how the amount of simple and normal data impacts perplexity, Figure 2 shows perplexity scores for models trained on varying amounts of simple data as we add increasing amounts of normal data. We again see that normal data is beneficial; regardless of the amount of simple data, adding normal data improves perplexity. This improvement is most beneficial when simple data is limited. Models trained on less simple data achieved larger performance increases than those models trained on more simple data.

Figure 2 also shows again that simple data is more valuable than normal data. For example, the simple-only model trained on 250K sentences achieves a perplexity of approximately 150. To achieve this same perplexity level starting with 200K simple sentences requires an additional 300K normal sentences, or starting with 100K simple sentences an additional 850K normal sentences.

### 4.3 Language Model Adaptation

In the experiments above, we generated the language models by treating the simple and normal data as one combined corpus. This approach has the benefit of simplicity, however, better performance for combining related corpora has been seen by domain adaptation techniques which combine the data in more structured ways (Bacchiani and Roark, 2003). *Our goal for this paper is not to explore domain adaptation techniques, but to determine if normal data is useful for the simple language modeling task.* However, to provide another dimension for comparison and to understand

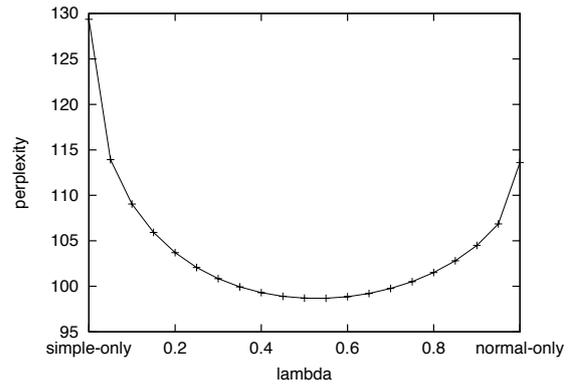


Figure 3: Perplexity scores for a linearly interpolated model between the simple-only model and the normal-only model for varying lambda values.

if domain adaptation techniques may be useful, we also investigated a linearly interpolated language model.

A linearly interpolated language model combines the probabilities of two or more language models as a weighted sum. In our case, the interpolated model combines the simple model estimate,  $p_s(w_i|w_{i-2}, w_{i-1})$ , and the normal model estimate,  $p_n(w_i|w_{i-2}, w_{i-1})$ , linearly (Jelinek and Mercer, 1980; Hsu, 2007):

$$p_{interpolated}(w_i|w_{i-2}, w_{i-1}) = \lambda p_n(w_i|w_{i-2}, w_{i-1}) + (1 - \lambda) p_s(w_i|w_{i-2}, w_{i-1})$$

where  $0 \leq \lambda \leq 1$ .

Figure 3 shows perplexity scores for varying lambda values ranging from the simple-only model on the left with  $\lambda = 0$  to the normal-only model on the right with  $\lambda = 1$ . As with the previous experiments, adding normal data improves perplexity. In fact, with a lambda of 0.5 (equal weight between the models) the performance is slightly better than the aggregate approaches above with a perplexity of 98. The results also highlight the balance between simple and normal data; normal data is not as good as simple data and adding too much of it can cause the results to degrade.

## 5 Language Model Evaluation: Lexical Simplification

Currently, no automated methods exist for evaluating sentence-level or document-level text simplification systems and manual evaluation is time-consuming, expensive and has not been validated. Because of these evaluation challenges, we chose to evaluate the language models extrinsi-

Word:	<i>tight</i>
Context:	With the physical market as <b>tight</b> as it has been in memory, silver could fly at any time.
Candidates:	constricted, pressurised, low, high-strung, tight
Human ranking:	tight, low, constricted, pressurised, high-strung

Figure 4: A lexical substitution example from the SemEval 2012 data set.

cally based on the lexical simplification task from SemEval 2012 (Specia et al., 2012).

Lexical simplification is a sub-problem of the general text simplification problem (Chandrasekar and Srinivas, 1997); a sentence is simplified by substituting words or phrases in the sentence with “simpler” variations. Lexical simplification approaches have been shown to improve the readability of texts (Urano, 2000; Leroy et al., 2012), are useful in domains such as medical texts where major content changes are restricted, and they may be useful as a pre- or post-processing step for general simplification systems.

## 5.1 Experimental Setup

Examples from the lexical simplification data set from SemEval 2012 consist of three parts:  $w$ , the word to be simplified;  $s_1, \dots, s_{i-1}, w, s_{i+1}, \dots, s_n$ , a sentence containing the word; and,  $r_1, r_2, \dots, r_m$ , a list of candidate simplifications for  $w$ . The goal of the task is to rank the candidate simplifications according to their simplicity in the context of the sentence. Figure 4 shows an example from the data set. The data set contains a development set of 300 examples and a test set of 1710 examples.<sup>3</sup> For our experiments, we evaluated the models on the test set.

Given a language model  $p(\cdot)$  and a lexical simplification example, we ranked the list of candidates based on the probability the language model assigns to the sentence with the candidate simplification inserted in context. Specifically, we scored each candidate simplification  $r_j$  by

$$p(s_1 \dots s_{i-1} r_j s_{i+1} \dots s_n)$$

and then ranked them based on this score. For example, to calculate the ranking for the example in Figure 4 we calculate the probability of each of:

- With the physical market as *constricted* as it has been ...
- With the physical market as *pressurised* as it has been ...
- With the physical market as *low* as it has been ...
- With the physical market as *high-strung* as it has been ...
- With the physical market as *tight* as it has been ...

with the language model and then rank them by their probability. *We do not suggest this as a com-*

<sup>3</sup><http://www.cs.york.ac.uk/semeval-2012/task1/>

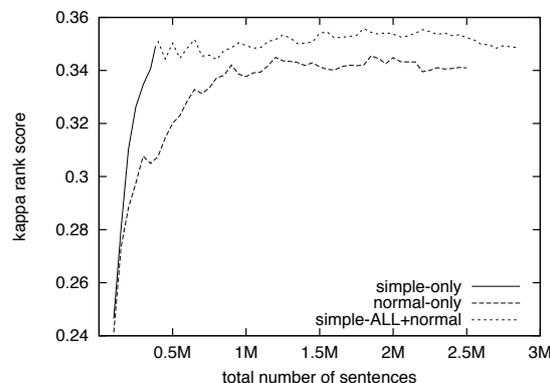


Figure 5: Kappa rank scores for the models trained on increasing amounts of data.

plete lexical substitution system, but it was a common feature for many of the submitted systems, it performs well relative to the other systems, and it allows for a concrete comparison between the language models on a simplification task.

To evaluate the rankings, we use the metric from the SemEval 2012 task, the Cohen’s kappa coefficient (Landis and Koch, 1977) between the system ranking and the human ranking, which we denote the “kappa rank score”. See Specia et al. (2012) for the full details of how the evaluation metric is calculated.

We use the same setup for training the language models as in the perplexity experiments except the models are open vocabulary instead of closed. Open vocabulary models allow for the language models to better utilize the varying amounts of data and since the lexical simplification problem only requires a comparison of probabilities within a given model to produce the final ranking, we do not need the closed vocabulary requirement.

## 5.2 Lexical Simplification Results

Figure 5 shows the kappa rank scores for the simple-only, normal-only and combined models. As with the perplexity results, for similar amounts of data the simple-only model performs better than the normal-only model. We also again see that the performance difference between the two models grows as the amount of data increases. However,

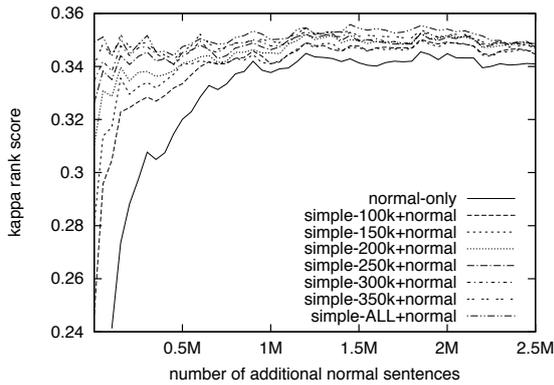


Figure 6: Kappa rank scores for models trained with varying amounts of simple data combined with increasing amounts of normal data.

unlike the perplexity results, simply appending additional normal data to the entire simple data set does not improve the performance of the lexical simplifier.

To determine if additional normal data improves the performance for models trained on smaller amounts of simple data, Figure 6 shows the kappa rank scores for models trained on different amounts of simple data as additional normal data is added. For smaller amounts of simple data adding normal data does improve the kappa rank score. For example, a language model trained with 100K simple sentences achieves a score of 0.246 and is improved by almost 40% to 0.344 by adding all of the additional normal data. Even the performance of a model trained with 300K simple sentences is increased by 3% (0.01 improvement in kappa rank score) by adding normal data.

### 5.3 Language Model Adaptation

The results in the previous section show that adding normal data to a simple data set can improve the lexical simplifier if the amount of simple data is limited. To investigate this benefit further, we again generated linearly interpolated language models between the simple-only model and the normal-only model. Figure 7 shows results for the same experimental design as Figure 6 with varying amounts of simple and normal data, however, rather than appending the normal data we trained the models separately and created a linearly interpolated model as described in Section 4.3. The best lambda was chosen based on a linear search optimized on the SemEval 2012 development set.

For all starting amounts of simple data, interpo-

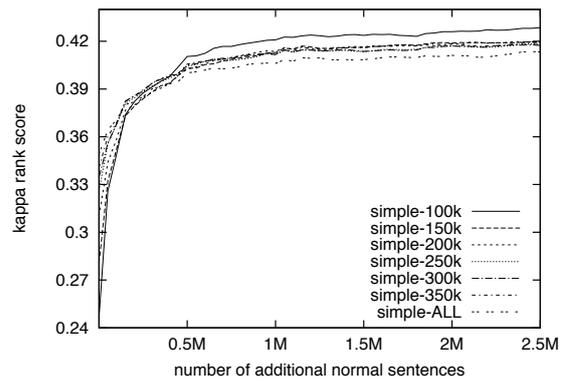


Figure 7: Kappa rank scores for linearly interpolated models between simple-only and normal-only models trained with varying amounts of simple and normal data.

lating the simple model with the normal model results in a large increase in the kappa rank score. *Combining the model trained on all the simple data with the model trained on all the normal data achieves a score of 0.419, an improvement of 23% over the model trained on only simple data.* Although our goal was not to create the best lexical simplification system, this approach would have ranked 6th out of 11 submitted systems in the SemEval 2012 competition (Specia et al., 2012).

Interestingly, although the performance of the simple-only models varied based on the amount of simple data, when these models are interpolated with a model trained on normal data, the performance tended to converge. This behavior is also seen in Figure 6, though to a lesser extent. This may indicate that for some tasks like lexical simplification, only a modest amount of simple data is required when combining with additional normal data to achieve reasonable performance.

## 6 Why Does Unsimplified Data Help?

For both the perplexity experiments and the lexical simplification experiments, utilizing additional normal data resulted in large performance improvements; using **all** of the simple data available, performance is still significantly improved when combined with normal data. In this section, we investigate why the additional normal data is beneficial for simple language modeling.

### 6.1 More $n$ -grams

Intuitively, adding normal data provides additional English data to train on. Most language models are

	Perplexity test data			Lexical simplification		
	simple	normal	% inc.	simple	normal	% inc.
1-grams	0.85	0.93	9.4%	0.74	0.78	6.2%
2-grams	0.66	0.82	24%	0.34	0.54	56%
3-grams	0.39	0.57	46%	0.088	0.19	117%

Table 3: Proportion of  $n$ -grams in the test sets that occur in the simple and normal training data sets.

trained using a smoothed version of the maximum likelihood estimate for an  $n$ -gram. For trigrams, this is:

$$p(a|bc) = \frac{\text{count}(abc)}{\text{count}(bc)}$$

where  $\text{count}(\cdot)$  is the number of times the  $n$ -gram occurs in the training corpus. For interpolated and backoff  $n$ -gram models, these counts are smoothed based on the probabilities of lower order  $n$ -gram models, which are in-turn calculated based on counts from the corpus.

We hypothesize that the key benefit of additional normal data is access to more  $n$ -gram counts and therefore better probability estimation, particularly for  $n$ -grams in the simple corpus that are unseen or have low frequency. For  $n$ -grams that have never been seen before, the normal data provides some estimate from English text. This is particularly important for unigrams (i.e. words) since there is no lower order model to gain information from and most language models assume a uniform prior on unseen words, treating them all equally. For  $n$ -grams that have been seen but are rare, the additional normal data can help provide better probability estimates. Because frequencies tend to follow a Zipfian distribution, these rare  $n$ -grams make up a large portion of  $n$ -grams in real data (Ha et al., 2003).

To partially validate this hypothesis, we examined the  $n$ -gram overlap between the  $n$ -grams in the training data and the  $n$ -grams in the test sets from the two tasks. Table 3 shows the percentage of unigrams, bigrams and trigrams from the two test sets that are found in the simple and normal training data.

For all  $n$ -gram sizes the normal data contained more test set  $n$ -grams than the simple data. Even at the unigram level, the normal data contained significantly more of the test set unigrams than the simple data. On the perplexity data set, the 9.4% increase in word occurrence between the simple and normal data set represents an over 50% reduction in the number of out of vocabulary words. For

	Perplexity test data		Lexical simplification	
	simple + normal	% inc. over normal	simple + normal	% inc. over normal
1-grams	0.93	0.2%	0.78	0.0%
2-grams	0.83	0.8%	0.54	1.1%
3-grams	0.58	2.5%	0.20	2.6%

Table 4: Proportion of  $n$ -grams in the test sets that occur in the *combination* of both the simple and normal data.

larger  $n$ -grams, the difference between the simple and normal data sets are even more pronounced. On the lexical simplification data the normal data contained more than twice as many test trigrams as the simple data. These additional  $n$ -grams allow for better probability estimates on the test data and therefore better performance on the two tasks.

## 6.2 The Role of Normal Data

Estimation of rare events is one component of language model performance, but other factors also impact performance. Table 4 shows the test set  $n$ -gram overlap on the combined data set of simple and normal data. Because the simple and normal data come from the same content areas, the simple data provides little additional coverage if the normal data is already used. For example, adding the simple data to the normal data only increases the number of seen unigrams by 0.2%, representing only about 600 new words. However, the experiments above showed the combined models performed much better than models trained only on normal data.

This discrepancy highlights the key problem with normal data: it is out-of-domain data. While it shares some characteristics with the simple data, it represents a different distribution over the language. To make this discrepancy more explicit, we created a sentence aligned data set by aligning the simple and normal articles using the approach from Coster and Kauchak (2011b). This approach has been previously used for aligning English Wikipedia and Simple English Wikipedia with reasonable accuracy. The resulting data set contains 150K aligned simple-normal sentence pairs.

Figure 8 shows the perplexity scores for language models trained on this data set. Because the data is aligned and therefore similar, we see the perplexity curves run parallel to each other as more data is added. However, even though these

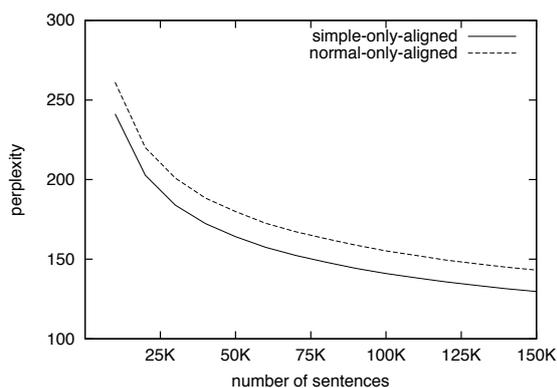


Figure 8: Language model perplexities for models trained on increasing data sizes for a simple-normal sentence aligned data set.

sentences represent the same content, the language use is different between simple and normal and the normal data performs consistently worse.

### 6.3 A Balance Between Simple and Normal

Examining the optimal lambda values for the linearly interpolated models also helps understand the role of the normal data. On the perplexity task, the best perplexity results were obtained with a lambda of 0.5, or an equal weighting between the simple and normal models. Even though the normal data contained six times as many sentences and nine times as many words, the best modeling performance balanced the quality of the simple model with the coverage of the normal model.

For the simplification task, the optimal lambda value determined on the development set was 0.98, with a very strong bias towards the simple model. Only when the simple model did not provide differentiation between lexical choices will the normal model play a role in selecting the candidates. For the lexical simplification task, the role of the normal model is even more clear: to handle rare occurrences not covered by the simple model and to smooth the simple model estimates.

## 7 Conclusions and Future Work

In the experiments above we have shown that on two different tasks utilizing additional normal data improves the performance of simple English language models. On the perplexity task, the combined model achieved a performance improvement of 23% over the simple-only model and on the lexical simplification task, the combined model achieved a 24% improvement. These improve-

ments are achieved over a simple-only model that uses *all* simple English data currently available in this domain.

For both tasks, the best improvements were seen when using language model adaptation techniques, however, the adaptation results also indicated that the role of normal data is partially task dependent. On the perplexity task, the best results were achieved with an equal weighting between the simple-only and normal-only model. However, on the lexical simplification task, the best results were achieved with a very strong bias towards the simple-only model. For other simplification tasks, the optimal parameters will need to be investigated.

For many of the experiments, combining a smaller amount of simple data (50K-100K sentences) with normal data achieved results that were similar to larger simple data set sizes. For example, on the lexical simplification task, when using a linearly interpolated model, the model combining 100K simple sentences with all the normal data achieved comparable results to the model combining all the simple sentences with all the normal data. This is encouraging for other monolingual domains such as text compression or text simplification in non-English languages where less data is available.

There are still a number of open research questions related to simple language modeling. First, further experiments with larger normal data sets are required to understand the limits of adding out-of-domain data. Second, we have only utilized data from Wikipedia for normal text. Many other text sources are available and the impact of not only size, but also of domain should be investigated. Third, it still needs to be determined how language model performance will impact sentence-level and document-level simplification approaches. In machine translation, improved language models have resulted in significant improvements in translation performance (Brants et al., 2007). Finally, in this paper we only investigated linearly interpolated language models. Many other domain adaptations techniques exist and may produce language models with better performance.

## References

- Michiel Bacchiani and Brian Roark. 2003. Unsupervised language model adaptation. In *Proceedings of ICASSP*.
- Michele Banko, Vibhu Mittal, and Michael Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of ACL*.
- Jerome R. Bellegarda. 2004. Statistical language model adaptation: Review and perspectives. *Speech Communication*.
- Or Biran, Samuel Brody, and Noémie Elhadad. 2011. Putting it simply: A context-aware approach to lexical simplification. In *Proceedings of ACL*.
- Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Proceedings of EMNLP*.
- Raman Chandrasekar and Bangalore Srinivas. 1997. Automatic induction of rules for text simplification. *Knowledge Based Systems*.
- Stanley Chen, Douglas Beeferman, and Ronald Rosenfeld. 1998. Evaluation metrics for language models. In *DARPA Broadcast News Transcription and Understanding Workshop*.
- Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*.
- William Coster and David Kauchak. 2011a. Learning to simplify sentences using Wikipedia. In *Proceedings of Text-To-Text Generation*.
- William Coster and David Kauchak. 2011b. Simple English Wikipedia: A new text simplification task. In *Proceedings of ACL*.
- Hal Daume and Daniel Marcu. 2002. A noisy-channel model for document compression. In *Proceedings of ACL*.
- Christopher Cieri David Graff. 2003. English gigaword. <http://www ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T05>.
- Carsten Eickhoff, Pavel Serdyukov, and Arjen P. de Vries. 2010. Web page classification on child suitability. In *Proceedings of CIKM*.
- Michel Galley and Kathleen McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Proceedings of HLT-NAACL*.
- Le Quan Ha, E. I. Sicilia-Garcia, Ji Ming, and F. J. Smith. 2003. Extension of Zipf's law to word and character  $n$ -grams for English and Chinese. *Computational Linguistics and Chinese Language Processing*.
- Bo-June Hsu. 2007. Generalized linear interpolation of language models. In *IEEE Workshop on ASRU*.
- Frederick Jelinek and Robert Mercer. 1980. Interpolated estimation of markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice*.
- Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artificial Intelligence*.
- J. Richard Landis and Gary G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*.
- Gondy Leroy, James E. Endicott, Obay Mouradi, David Kauchak, and Melissa Just. 2012. Improving perceived and actual text difficulty for health information consumers using semi-automated methods. In *American Medical Informatics Association (AMIA) Fall Symposium*.
- Courtney Napoles and Mark Dredze. 2010. Learning simple Wikipedia: A cogitation in ascertaining abecedarian language. In *Proceedings of HLT/NAACL Workshop on Computation Linguistics and Writing*.
- Tadashi Nomoto. 2009. A comparison of model free versus model intensive approaches to sentence compression. In *Proceedings of EMNLP*.
- Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*.
- Ronald Rosenfeld. 1996. A maximum entropy approach to adaptive statistical language modeling. *Computer, Speech and Language*.
- Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. Semeval-2012 task 1: English lexical simplification. In *Joint Conference on Lexical and Computational Semantics (\*SEM)*.
- Lucia Specia. 2010. Translating from complex to simplified sentences. In *Proceedings of Computational Processing of the Portuguese Language*.
- Andreas Stolcke. 2002. SRILM - An extensible language modeling toolkit. In *Proceedings of ICSLP*.
- Hisami Suzuki and Jianfeng Gao. 2005. A comparative study on language model adaptation techniques. In *Proceedings of EMNLP*.
- Jenine Turner and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of ACL*.
- Ken Urano. 2000. Lexical simplification and elaboration: Sentence comprehension and incidental vocabulary acquisition. Master's thesis, University of Hawaii.

- Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of EMNLP*.
- Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of ACL*.
- Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Proceedings of NAACL*.
- Bing Zhao, Matthias Eck, and Stephan Vogel. 2004. Language model adaptation for statistical machine translation with structured query models. In *Proceedings of COLING*.
- Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of ICCL*.

# Combining Referring Expression Generation and Surface Realization: A Corpus-Based Investigation of Architectures

Sina Zarriß      Jonas Kuhn

Institut für maschinelle Sprachverarbeitung

University of Stuttgart, Germany

sina.zarriess, jonas.kuhn@ims.uni-stuttgart.de

## Abstract

We suggest a generation task that integrates discourse-level referring expression generation and sentence-level surface realization. We present a data set of German articles annotated with deep syntax and referents, including some types of implicit referents. Our experiments compare several architectures varying the order of a set of trainable modules. The results suggest that a revision-based pipeline, with intermediate linearization, significantly outperforms standard pipelines or a parallel architecture.

## 1 Introduction

Generating well-formed linguistic utterances from an abstract non-linguistic input involves making a multitude of conceptual, discourse-level as well as sentence-level, lexical and syntactic decisions. Work on rule-based natural language generation (NLG) has explored a number of ways to combine these decisions in an architecture, ranging from integrated systems where all decisions happen jointly (Appelt, 1982) to strictly sequential pipelines (Reiter and Dale, 1997). While integrated or interactive systems typically face issues with efficiency and scalability, they can directly account for interactions between discourse-level planning and linguistic realization. For instance, Rubinfoff (1992) mentions Example (1) where the sentence planning component needs to have access to the lexical knowledge that “order” and not “home” can be realized as a verb in English.

- (1)    a.    \*John homed him with an order.  
      b.    John ordered him home.

In recent data-driven generation research, the focus has somewhat shifted from full data-to-text systems to approaches that isolate well-defined

subproblems from the NLG pipeline. In particular, the tasks of surface realization and referring expression generation (REG) have received increasing attention using a number of available annotated data sets (Belz and Kow, 2010; Belz et al., 2011). While these single-task approaches have given rise to many insights about algorithms and corpus-based modelling for specific phenomena, they can hardly deal with aspects of the architecture and interaction between generation levels.

This paper suggests a middle ground between full data-to-text and single-task generation, combining two well-studied NLG problems. We integrate a discourse-level approach to REG with sentence-level surface realization in a data-driven framework. We address this integrated task with a set of components that can be trained on flexible inputs which allows us to systematically explore different ways of arranging the components in a generation architecture. Our main goal is to investigate how different architectural set-ups account for interactions between generation decisions at the level of referring expressions (REs), syntax and word order.

Our basic set-up is inspired from the Generating Referring Expressions in Context (GREC) tasks, where candidate REs have to be assigned to instances of a referent in a Wikipedia article (Belz and Kow, 2010). We have created a dataset of German texts with annotations that extend this standard in three substantial ways: (i) our domain consists of articles about *robbery* events that mainly involve two main referents, a *victim* and a *perpetrator* (*perp*), (ii) annotations include deep and shallow syntactic relations similar to the representations used in (Belz et al., 2011) (iii) annotations include empty referents, as e.g. in passives and nominalizations directing attention to the phenomenon of implicit reference, which is largely understudied in NLG. Figure 1 presents an example for a deep syntax tree with underspecified RE

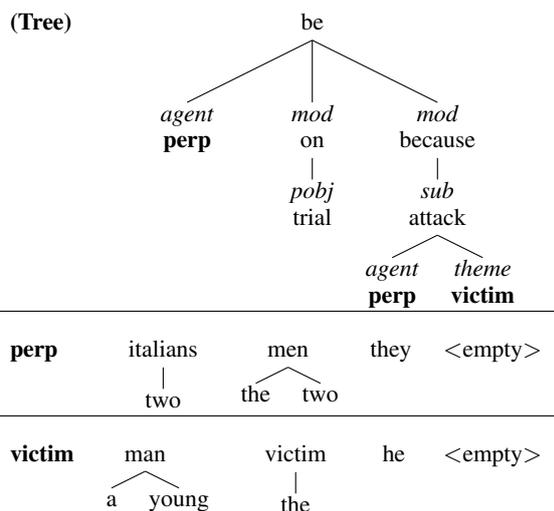


Figure 1: Underspecified tree with RE candidates

slots and lists of candidates REs for each referent.

Applying a strictly sequential pipeline on our data, we observe incoherent system output that is related to an interaction of generation levels, very similar to the interleaving between sentence planning and lexicalization in Example (1). A pipeline that first inserts REs into the underspecified tree in Figure 1, then generates syntax and finally linearizes, produces inappropriate sentences like (2-a).

- (2) a. \*[The two men]<sub>p</sub> are on trial because of an attack by [two italians]<sub>p</sub> on [a young man]<sub>v</sub>.  
 b. [Two italians]<sub>p</sub> are on trial because of an attack on [a young man]<sub>v</sub>.

Sentence (2-a) is incoherent because the syntactic surface obscures the intended meaning that “two italians” and “the two men” refer to the same referent. In order to generate the natural Sentence (2-b), the RE component needs information about linear precedence of the two *perp* instances and the nominalization of “attack”. These types of interactions between referential and syntactic realization have been thoroughly discussed in theoretical accounts of textual coherence, as e.g. Centering Theory (Grosz et al., 1995).

The integrated modelling of REG and surface realization leads to a considerable expansion of the choice space. In a sentence with 3 referents that each have 10 RE candidates and can be freely ordered, the number of surface realizations increases from 6 to  $6 \cdot 10^3$ , assuming that the remaining words can not be syntactically varied. Thus, even when the generation problem is restricted to these tasks, a fully integrated architecture faces scalability issues on realistic corpus data.

In this work, we assume a modular set-up of the generation system that allows for a flexible ordering of the single components. Our experiments vary 3 parameters of the generation architecture: 1) the sequential order of the modules, 2) parallelization of modules, 3) joint vs. separate modelling of implicit referents. Our results suggest that the interactions between RE and syntax can be modelled in sequential generation architecture where the RE component has access to information about syntactic realization and an approximative, intermediate linearization. Such a system is reminiscent of earlier work in rule-based generation that implements an interactive or revision-based feedback between discourse-level planning and linguistic realisation (Hovy, 1988; Robin, 1993).

## 2 Related Work

Despite the common view of NLG as a pipeline process, it is a well-known problem that high-level, conceptual knowledge and low-level linguistic knowledge are tightly interleaved (Danlos, 1984; Mellish et al., 2000). In rule-based, strictly sequential generators these interactions can lead to a so-called *generation gap*, where a downstream module cannot realize a text or sentence plan generated by the preceding modules (Meteer, 1991; Wanner, 1994). For this reason, a number of other architectures has been proposed, see De Smedt et al. (1996) for an overview. For reasons of tractability and scalability, many practical NLG systems still have been designed as sequential pipelines that follow the basic layout of macroplanning-microplanning-linguistic realization (Reiter, 1994; Cahill et al., 1999; Bateman and Zock, 2003).

In recent data-driven research on NLG, many single tasks have been addressed with corpus-based methods. For surface realization, the standard set-up is to regenerate from syntactic representations that have been produced for realistic corpus sentences. The first widely known statistical approach by Langkilde and Knight (1998) used language-model n-gram statistics on a word lattice of candidate realisations to guide a ranker. Subsequent work explored ways of exploiting linguistically annotated data for trainable generation models (Ratnaparkhi, 2000; Belz, 2005). Work on data-driven approaches has led to insights about the importance of linguistic features for sentence

linearization decisions (Ringger et al., 2004; Filipova and Strube, 2007; Cahill and Riester, 2009). (Zarriß et al., 2012) have recently argued that the good performance of these linguistically motivated word order models, which exploit morpho-syntactic features of noun phrases (i.e. referents), is related to the fact that these morpho-syntactic features implicitly encode a lot of knowledge about the underlying discourse or information structure.

A considerable body of REG research has been done in the paradigm established by Dale (1989; 1995). More closely related to our work are approaches in the line of Siddharthan and Copestake (2004) or Belz and Varges (2007) who generate contextually appropriate REs for instances of a referent in a text. Belz and Varges (2007)’s GREC data set includes annotations of implicit subjects in coordinations. Zarriß et al. (2011) deal with implicit subjects in passives, proposing a set of heuristics for adding these agents to the generation input. Roth and Frank (2012) acquire automatic annotations of implicit roles for the purpose of studying coherence patterns in texts. Implicit referents have also received attention for the analysis of semantic roles (Gerber and Chai, 2010; Ruppenhofer et al., 2010).

Statistical methods for data-to-text generation have been explored only recently. Belz (2008) trains a probabilistic CFG to generate weather forecasts, Chen et al. (2010) induce a synchronous grammar to generate sportcaster text. Both address a restricted domain where a direct alignment between units in the non-linguistic representation and the linguistic utterance can be learned. Marciniak and Strube (2005) propose an ILP model for global optimization in a generation task that is decomposed into a set of classifiers. Bohnet et al. (2011) deal with multi-level generation in a statistical framework and in a less restricted domain. They adopt a standard sequential pipeline approach.

Recent corpus-based generation approaches faced the problem that existing standard treebank representations for parsing or other analysis tasks do not necessarily fit the needs of generation (Bohnet et al., 2010; Wanner et al., 2012). Zarriß et al. (2011) discuss the problem of an input representation that is appropriately underspecified for the realistic generation of voice alternations.

### 3 The Data Set

The data set for our generation experiments consists of 200 newspaper articles about robbery events. The articles were extracted from a large German newspaper corpus. A complete example text with RE annotations is given in Figure 2, Table 1 summarizes some data set statistics.

#### 3.1 RE annotation

The RE annotations mark explicit and implicit mentions of referents involved in the robbery event described in an article. Explicit mentions are marked as spans on the surface sentence, labeled with the referent’s role and an ID. We annotate the following referential roles: (i) *perpetrator* (*perp*), (ii) *victim*, (iii) *source*, according to the core roles of the *Robbery* frame in English FrameNet. We include *source* since some texts do not mention a particular *victim*, but rather the location of the robbery (e.g. a bank, a service station). The ID distinguishes referents that have the same role, e.g. “the husband” and the “young family” in Sentences (3-a) and (3-d) in Figure 2. Each RE is linked to its syntactic head. This complies with the GREC data sets, and is also useful for further annotation of the deep syntax level (see Section 3.2).

The RE implicit mentions of *victim*, *perp*, and *source* are annotated as attributes of their syntactic heads in the surface sentence. We consider the following types of implicit referents: (i) agents in passives (e.g. “robbed” in (3-a)), (ii) arguments of nominalizations (e.g. “resistance” in (3-e)), (iii) possessives (e.g. “watch” in (3-f)), (iv) missing subjects in coordinations. (e.g. “flee” in (3-f))

The brat tool (Stenetorp et al., 2012) was used for annotation. We had 2 annotators with a computational linguistic background, provided with annotation guidelines. They were trained on a set of 20 texts. We measure a good agreement on another set of 15 texts: the simple pairwise agreement for explicit mentions is 95.14%-96.53% and 78.94%-76.92% for implicit mentions.<sup>1</sup>

#### 3.2 Syntax annotation

The syntactic annotation of our data includes two layers: shallow and deep, labeled dependencies, similar to the representation used in surface realization shared tasks (Belz et al., 2011). We use

<sup>1</sup>Standard measures for the “above chance annotator agreement” are only defined for task where the set of annotated items is pre-defined.

- (3) a. Junge Familie<sub>v:0</sub> auf dem Heimweg<sub>poss:v</sub> ausgeraubt<sub>ag:p</sub>  
Young family on the way home<sub>poss:v</sub> robbed<sub>ag:p</sub>
- b. Die Polizei sucht nach zwei ungepflegt wirkenden jungen Männern im Alter von etwa 25 Jahren<sub>p:0</sub>.  
The police looks for two shabby-looking young men of about 25 years.
- c. Sie<sub>p:0</sub> sollen am Montag gegen 20 Uhr eine junge Familie mit ihrem sieben Monate alten Baby<sub>v:0</sub> auf  
They are said to on Monday around 20 o'clock a young family with their seven month old baby on  
dem Heimweg<sub>poss:v</sub> von einem Einkaufsbummel überfallen und ausgeraubt haben.  
the way home<sub>poss:v</sub> from a shopping tour attacked and robbed have.
- d. Wie die Polizei berichtet, drohten die zwei Männer<sub>p:0</sub> dem Ehemann<sub>v:1</sub>, ihn<sub>v:1</sub> zusammenschlagen.  
As the police reports, threatened the two men the husband him beat up.
- e. Er<sub>v:1</sub> gab deshalb seine<sub>v:1</sub> Brieftasche ohne Gegenwehr<sub>ag:v,the:p</sub> heraus.  
He gave therefore his wallet without resistance<sub>ag:v,the:p</sub> out.
- f. Anschließend nahmen ihm<sub>v:1</sub> die Räuber<sub>p:0</sub> noch die Armbanduhr<sub>poss:v</sub> ab und flüchteten<sub>ag:p</sub>.  
Afterwards took him the robbers also the watch<sub>poss:v</sub> off and fleed<sub>ag:p</sub>.

Figure 2: Example text with RE annotations, oval boxes mark *victim* mentions, square boxes mark *perp* mentions, heads of implicit arguments are underlined

the Bohnet (2010) dependency parser to obtain an automatic annotation of shallow or surface dependencies for the corpus sentences.

The deep syntactic dependencies are derived from the shallow layer by a set of hand-written transformation rules. The goal is to link referents to their main predicate in a uniform way, independently of the surface-syntactic realization of the verb. We address passives, nominalizations and possessives corresponding to the contexts where we annotated implicit referents (see above). The transformations are defined as follows:

1. remove auxiliary nodes, verb morphology and finiteness, a tense feature distinguishes past and present, e.g. “haben:AUX überfallen:VVINF” (*have attacked*) maps to “überfallen:VV:PAST” (*attack:PAST*)
2. map subjects in actives and oblique agents in passives to “agents”; objects in actives and subjects in passive to “themes”, e.g. *victim/subj was attacked by perp/obl-ag* maps to *perp/agent attack victim/theme*
3. attach particles to verb lemma, e.g. “gab” ... “heraus” in (3-e) is mapped to “herausgeben” (*give to*)
4. map nominalized to verbal lemmas, their prepositional and genitive arguments to semantic subjects and objects, e.g. *attack on victim* is mapped to *attack victim/theme*
5. normalize prenominal and genitive postnominal possessives, e.g. “seine Brieftasche” (*his wallet*) and “die Brieftasche des Opfers” (*the wallet of the victim*) map to “die Brieftasche POSS victim” (*the wallet of victim*), only applies if possessive is an annotated RE

Nominalizations are mapped to their verbal base forms on the basis of lexicalized rules for the nominalized lemmas observed in the corpus. The other transformations are defined on the shallow dependency annotation.

# sentences	2030
# explicit REs	3208
# implicit REs	1778
# passives	383
# nominalizations	393
# possessives	1150

Table 1: Basic annotation statistics

### 3.3 Multi-level Representation

In the final representation of our data set, we integrate the RE and deep syntax annotation by replacing subtrees corresponding to an RE span. The RE slot in the tree of the sentence is labeled with its referential role and its ID. All RE subtrees for a referent in a text are collected in a candidate list which is initialized with three default REs: (i) a pronoun, (ii) a default nominal (e.g. “the victim”), (iii) the empty RE. In contrast to the GREC data sets, our RE candidates are not represented as the original surface strings, but as non-linearized subtrees. The resulting multi-layer representation for each text is structured as follows:

1. unordered deep trees with RE slots (*deepSyn<sub>-re</sub>*)
2. unordered shallow trees with RE slots (*shallowSyn<sub>-re</sub>*)
3. unordered RE subtrees
4. linearized, fully specified surface trees (*linSyn<sub>+re</sub>*)
5. alignments between nodes in 1., 2., 4.

The generation components in Section 4 also use intermediate layers where REs are inserted into the deep trees (*deepSyn<sub>+re</sub>*) or shallow trees (*shallowSyn<sub>+re</sub>*).

Nodes in unordered trees are deterministically sorted by their : 1. distance to the root, 2. label,

3. PoS tag, 4. lemma. The generation components traverse the nodes in this the order.

## 4 Generation Systems

Our main goal is to investigate different architectures for combined surface realization and referring expression generation. We assume that this task is split into three main modules: a syntax generator, an REG component, and a linearizer. The components are implemented in a way that they can be trained and applied on varying inputs, depending on the pipeline. Section 4.1 describes the basic set-up of our components. Section 4.2 defines the architectures that we will compare in our experiments (Section 5). Section 4.3 presents the implementation of the underlying feature models.

### 4.1 Components

#### 4.1.1 SYN: Deep to Shallow Syntax

For mapping deep to shallow dependency trees, the syntax generator induces a probabilistic tree transformation. The transformations are restricted to verb nodes in the deep tree (possessives are handled in the RE module) and extracted from the alignments between the deep and shallow layer in the training input. As an example, the deep node “attack:VV” aligns to “have:AUX attacked:VVINF”, “attacks:VVFİN”, “the:ART attack:NN on:PRP”. The learner is implemented as a ranking component, trained with SVMrank (Joachims, 2006). During training, each instance of a verb node has one optimal shallow dependency alignment and a set of distractor candidates. During testing, the module has to pick the best shallow candidate according to its feature model.

In our crossvalidation set-up (see Section 5), we extract, on average, 374 transformations from the training sets. This set subdivides into non-lexicalized and lexicalized transformations. The mapping rule in (4-a) that simply rewrites the verb underspecified PoS tag to the finite verb tag in the shallow tree illustrates the non-lexicalized case. Most transformation rules (335 out of 374 on average) are lexicalized for a specific verb lemma and mostly transform nominalizations as in rule (4-b) and particles (see Section 3.2).

- (4) a.  $(x, lemma, VV, y) \rightarrow (x, lemma, VVFİN, y)$   
b.  $(x, überfallen/attack, VV, y) \rightarrow (x, bei/at, PREP, y), (z, Überfall/attack, NN, x), (q, der/the, ART, z)$

The baseline for the verb transformation component is a two-step procedure: 1) pick a lexical-

ized rule if available for that verb lemma, 2) pick the most frequent transformation.

#### 4.1.2 REG: Realizing Referring Expressions

Similar to the syntax component, the REG module is implemented as a ranker that selects surface RE subtrees for a given referential slot in a deep or shallow dependency tree. The candidates for the ranking correspond to the entire set of REs used for that referential role in the original text (see Section 3.1). The basic RE module is a joint model of all RE types, i.e. nominal, pronominal and empty realizations of the referent. For the experiment in Section 5.4, we use an additional separate classifier for implicit referents, also trained with SVMrank. It uses the same feature model as the full ranking component, but learns a binary distinction for implicit or explicit mentions of a referent. The explicit mentions will be passed to the RE ranking component.

The baseline for the REG component is defined as follows: if the preceding and the current RE slot are instances of the same referent, realize a pronoun, else realize the longest nominal RE candidate that has not been used in the preceding text.

#### 4.1.3 LIN: Linearization

For linearization, we use the state-of-the-art dependency linearizer described in Bohnet et al. (2012). We train the linearizer on an automatically parsed version of the German TIGER treebank (Brants et al., 2002). This version was produced with the dependency parser by Bohnet (2010), trained on the dependency conversion of TIGER by Seeker and Kuhn (2012).

### 4.2 Architectures

Depending on the way the generation components are combined in an architecture, they will have access to different layers of the input representation. The following definitions of architectures recur to the layers introduced in Section 3.3.

#### 4.2.1 First Pipeline

The first pipeline corresponds most closely to a standard generation pipeline in the sense of (Reiter and Dale, 1997). REG is carried out prior to surface realization such that the RE component does not have access to surface syntax or word order whereas the SYN component has access to fully specified RE slots.

- training

1. train REG: ( $deepSyn_{-re}, deepSyn_{+re}$ )
2. train SYN: ( $deepSyn_{+re}, shallowSyn_{+re}$ )

- prediction
  1. apply REG:  $deepSyn_{-re} \rightarrow deepSyn_{+re}$
  2. apply SYN:  $deepSyn_{+re} \rightarrow shallowSyn_{+re}$
  3. linearize:  $shallowSyn_{+re} \rightarrow linSyn_{+re}$

#### 4.2.2 Second Pipeline

In the second pipeline, the order of the RE and SYN component is switched. In this case, REG has access to surface syntax without word order but the surface realization is trained and applied on trees with underspecified RE slots.

- training
  1. train SYN: ( $deepSyn_{-re}, shallowSyn_{-re}$ )
  2. train REG: ( $shallowSyn_{-re}, shallowSyn_{+re}$ )
- prediction
  1. apply SYN:  $deepSyn_{-re} \rightarrow shallowSyn_{-re}$
  2. apply REG:  $shallowSyn_{-re} \rightarrow shallowSyn_{+re}$
  3. linearize:  $shallowSyn_{+re} \rightarrow linSyn_{+re}$

#### 4.2.3 Parallel System

A well-known problem with pipeline architectures is the effect of error propagation. In our parallel system, the components are trained independently of each other and applied in parallel on the deep syntactic input with underspecified REs.

- training
  1. train SYN: ( $deepSyn_{-re}, shallowSyn_{-re}$ )
  2. train REG: ( $deepSyn_{-re}, deepSyn_{+re}$ )
- prediction
  1. apply REG and SYN:  $deepSyn_{-re} \rightarrow shallowSyn_{+re}$
  2. linearize:  $shallowSyn_{+re} \rightarrow linSyn_{+re}$

#### 4.2.4 Revision-based System

In the revision-based system, the RE component has access to surface syntax and a preliminary linearization, called *prelinSyn*. In this set-up, we apply the linearizer first on trees with underspecified RE slots. For this step, we insert the default REs for the referent into the respective slots. After REG, the tree is linearized once again.

- training
  1. train SYN on gold pairs of ( $deepSyn_{-re}, shallowSyn_{-re}$ )
  2. train REG on gold pairs of ( $prelinSyn_{-re}, prelinSyn_{+re}$ )
- prediction
  1. apply SYN:  $deepSyn_{-re} \rightarrow shallowSyn_{-re}$
  2. linearize:  $shallowSyn_{-re} \rightarrow prelinSyn_{-re}$
  3. apply REG:  $prelinSyn_{-re} \rightarrow prelinSyn_{+re}$
  4. linearize:  $prelinSyn_{+re} \rightarrow linSyn_{+re}$

### 4.3 Feature Models

The implementation of the feature models is based on a general set of templates for the SYN and REG component. The exact form of the models depends on the input layer of a component in a given architecture. For instance, when SYN is trained on  $deepSyn_{-re}$ , the properties of the children nodes are less specific for verbs that have RE slots as their dependents. When the SYN component is trained on  $deepSyn_{+re}$ , lemma and POS of the children nodes are always specified.

The feature templates for SYN combine properties of the shallow candidate nodes (label, PoS and lemma for top node and its children) with the properties of the instance in the tree: (i) lemma, tense, (ii) sentence is a header, (iii) label, PoS, lemma of mother node, children and grandchildren nodes (iv) number, lemmas of other verbs in the sentence.

The feature templates for REG combine properties of the candidate RE (PoS and lemma for top node and its children, length) with properties of the RE slot in the tree: lemma, PoS and labels for the (i) mother node, (ii) grandmother node, (iii) uncle and sibling nodes. Additionally, we implement a small set of global properties of a referent in a text: (i) identity is known, (ii) plural or singular referent, (iii) age is known, and a number of contextual properties capturing the previous referents and their predicted REs: (i) role and realization of the preceding referent, (ii) last mention of the current referent, (iii) realization of the referent in the header.

## 5 Experiments

In this experimental section, we provide a corpus-based evaluation of the generation components and architectures introduced in Section 4. In the following, Section 5.1 presents the details of our evaluation methodology. In Section 5.2, we discuss the first experiment that evaluates the pipeline architectures and the single components on oracle inputs. Section 5.3 describes an experiment which compares the parallel and the revision-based architecture against the pipeline. In Section 5.4, we compare two methods for dealing with the implicit referents in our data. Section 5.5 provides some general discussion of the results.

Input	System	Sentence overlap			SYN Accuracy		RE Accuracy		
		BLEU	NIST	BLEU <sub>r</sub>	String	Type	String	Type	Impl
<i>deepSyn<sub>-re</sub></i>	Baseline	42.38	9.9	47.94	35.66	44.81	33.3	36.03	50.43
<i>deepSyn<sub>-re</sub></i>	1st pipeline	54.65	11.30	59.95	57.09	68.15	54.61	71.51	84.72
<i>deepSyn<sub>-re</sub></i>	2nd pipeline	54.28	11.25	59.62	59.14	68.58	52.24	68.2	82
gold <i>deepSyn<sub>+re</sub></i>	SYN→LIN	63.9	12.7	62.86	60.83	69.74	100	100	100
gold <i>shallowSyn<sub>-re</sub></i>	REG→LIN	60.57	11.87	68.06	100	100	60.53	75.86	88.86
gold <i>shallowSyn<sub>+re</sub></i>	LIN	79.17	13.91	72.7	100	100	100	100	100

Table 2: Evaluating pipeline architectures against the baseline and upper bounds

## 5.1 Evaluation Measures

We split our data set into 10 splits of 20 articles. We use one split as the development set, and cross-validate on the remaining splits. In each case, the downstream modules of the pipeline will be trained on the jackknifed training set.

**Text normalization:** We carry out automatic evaluation calculated on lemmatized text without punctuation, excluding additional effects that would be introduced from a morphology generation component.

**Measures:** First, we use a number of evaluation measures familiar from previous generation shared tasks:

1. BLEU, sentence-level geometric mean of 1- to 4-gram precision, as in (Belz et al., 2011)
2. NIST, sentence-level n-gram overlap weighted in favour of less frequent n-grams, as in (Belz et al., 2011)
3. RE Accuracy on String, proportion of REs selected by the system with a string identical to the RE string in the original corpus, as in (Belz and Kow, 2010)
4. RE Accuracy on Type, proportion of REs selected by the system with an RE type identical to the RE type in the original corpus, as in (Belz and Kow, 2010)

Second, we define a number of measures motivated by our specific set-up of the task:

1. BLEU<sub>r</sub>, sentence-level BLEU computed on post-processed output where predicted referring expressions for *victim* and *perp* are replaced in the sentences (both gold and predicted) by their original role label, this score does not penalize lexical mismatches between corpus and system REs
2. RE Accuracy on Impl, proportion of REs predicted correctly as implicit/non-implicit
3. SYN Accuracy on String, proportion of shallow verb candidates selected by the system with a string identical to the verb string in the original corpus
4. SYN Accuracy on Type, proportion of shallow verb candidates selected by the system with a syntactic category identical to the category in the original corpus

## 5.2 Pipelines and Upper Bounds

The first experiment addresses the first and second pipeline introduced in Section 4.2.1 and 4.2.2. The baseline combines the baseline version of the SYN component (Section 4.1.1) and the REG component (Section 4.1.2) respectively. As we report in Table 2, both pipelines largely outperform the baseline. Otherwise, they obtain very similar scores in all measures with a small, weakly significant tendency for the first pipeline. The only remarkable difference is that the accuracy of the individual components is, in each case, lower when they are applied as the second step in the pipeline. Thus, the RE accuracy suffers from mistakes from the predicted syntax in the same way that the quality of syntax suffers from predicted REs.

The three bottom rows in Table 2 report the performance of the individual components and linearization when they are applied to inputs with an REG and SYN oracle, providing upper bounds for the pipelines applied on *deepSyn<sub>-re</sub>*. When REG and linearization are applied on *shallowSyn<sub>-re</sub>* with gold shallow trees, the BLEU score is lower (60.57) as compared to the system that applies syntax and linearization on *deepSyn<sub>+re</sub>*, deep trees with gold REs (BLEU score of 63.9). However, the BLEU<sub>r</sub> score, which generalizes over lexical RE mismatches, is higher for the REG→LIN components than for SYN→LIN. Moreover, the BLEU<sub>r</sub> score for the REG→LIN system comes close to the upper bound that applies linearization on *lin.Syn<sub>+re</sub>*, gold shallow trees with gold REs (BLEU<sub>r</sub> of 72.4), whereas the difference in standard BLEU and NIST is high. This effect indicates that the RE prediction mostly decreases BLEU due to lexical mismatches, whereas the syntax prediction is more likely to have a negative impact on final linearization.

The error propagation effects that we find in the first and second pipeline architecture clearly show that decisions at the levels of syntax, reference and word order interact, otherwise their predic-

Input	System	BLEU	NIST	BLEU <sub>r</sub>
<i>deepSyn-re</i>	1st pipeline	54.65	11.30	59.95
<i>deepSyn-re</i>	Parallel	54.78	11.30	60.05
<i>deepSyn-re</i>	Revision	56.31	11.42	61.30

Table 3: Architecture evaluation

tion would not affect each other. In particular, the REG module seems to be affected more seriously, the String Accuracy decreases from 60.53 on gold shallow trees to 52.24 on predicted shallow trees whereas the Verb String Accuracy decreases from 60.83 on gold REs to 57.04 on predicted REs.

### 5.3 Revision or parallelism?

The second experiment compares the first pipeline against the parallel and the revision-based architecture introduced in Section 4.2.3 and 4.2.4. The evaluation in Table 3 shows that the parallel architecture improves only marginally over the pipeline. By contrast, we obtain a clearly significant improvement for the revision-based architecture on all measures. The fact that this architecture significantly improves the BLEU, NIST and the BLEU<sub>r</sub> score of the parallel system indicates that the REG benefits from the predicted syntax when it is approximatively linearized. The fact that also the BLEU<sub>r</sub> score improves shows that a higher lexical quality of the REs leads to better final linearizations.

Table 4 shows the performance of the REG module on varying input layers, providing a more detailed analysis of the interaction between RE, syntax and word order. In order to produce the *deeplinSyn-re* layer, deep syntax trees with approximative linearizations, we preprocessed the deep trees by inserting a default surface transformation for the verb nodes. We compare this input for REG against the *prelinSyn-re* layer used in the revision-based architecture, and the *deepSyn-re* layer used in the pipeline and the parallel architecture. The REG module benefits from the linearization in the case of *deeplinSyn-re* and *prelinSyn-re*, outperforming the component trained applied on the non-linearized deep syntax trees. However, the REG module applied on *prelinSyn-re*, predicted shallow and linearized trees, clearly outperforms the module applied on *deeplinSyn-re*. This shows that the RE prediction can actually benefit from the predicted shallow syntax, but only when the predicted trees are approximatively linearized. As an upper bound, we report the performance obtained on

Input	System	RE Accuracy		
		String	Type	Impl
<i>deepSyn-re</i>	RE	54.61	71.51	84.72
<i>deeplinSyn-re</i>	RE	56.78	72.23	84.71
<i>prelinSyn-re</i>	RE	58.81	74.34	86.37
gold <i>linSyn-re</i>	RE	68.63	83.63	94.74

Table 4: RE generation from different input layers

*linSyn-re*, gold shallow trees with gold linearizations. This set-up corresponds to the GREC tasks. The gold syntax leads to a huge increase in performance.

These results strengthen the evidence from the previous experiment that decisions at the level of syntax, reference and word order are interleaved. A parallel architecture that simply “circumvents” error propagation effects by making decisions independent of each other is not optimal. Instead, the automatic prediction of shallow syntax can positively impact on RE generation if these shallow trees are additionally processed with an approximative linearization step.

### 5.4 A joint treatment of implicit referents?

The previous experiments have pursued a joint approach for modeling implicit referents. The hypothesis for this experiment is that the SYN component and the intermediate linearization in a revision-based architecture could benefit from a separate treatment of implicit referents since verb alternations like passive or nominalization often involve referent deletions.

The evaluation in Table 5 provides contradictory results depending on the evaluation measure. For the first pipeline, the system with a separate treatment of implicit referents significantly outperforms the joint system in terms of BLEU. However, the BLEU<sub>r</sub> score does not improve. In the revision-based architecture, we do not find a clear result for or against a joint modelling approach. The revision-based system with disjoint modelling of implicits shows a slight, non-significant increase in BLEU score. By contrast, the BLEU<sub>r</sub> score is significantly better for the joint approach. We experimented with parallelization of syntax generation and prediction of implicit referents in a revision-based system. This has a small positive effect on the BLEU<sub>r</sub> score and a small negative effect on the plain BLEU and NIST score. These contradictory scores might indicate that the automatic evaluation measures cannot capture all aspects of text quality, an issue that we discuss in the following.

- (5) Generated by sequential system:
- a. Deshalb gab dem Täter seine Brieftasche ohne daß das Opfer Widerstand leistet heraus.  
 Therefore gave to the robber his wallet without that the victim resistance shows out.
- b. Er nahm anschließend dem Opfer die Armbanduhr ab und der Täter flüchtete.  
 He takes afterwards the victim the watch off and the robber fled.
- (6) Generated by revision-based system:
- a. Das Opfer gibt deshalb seine Brieftasche ohne Widerstand zu leisten heraus.  
 The victim gave therefore his wallet without resistance to show out.
- b. Anschließend nahm der Täter dem Opfer die Armbanduhr ab und flüchtete.  
 Afterwards took the robber the victim the watch off and fled.

Figure 3: Two automatically generated outputs for the Sentences (3e-f) in Figure 2.

Joint	System	BLEU	NIST	BLEU <sub>r</sub>
+	1st pipeline	54.65	11.30	59.95
-	1st pipeline	55.38	11.48	59.52
+	Revision	56.31	11.42	61.30
-	Revision	56.42	11.54	60.52
-	Parallel+Revision	56.29	11.51	60.63

Table 5: Implicit reference and architectures

## 5.5 Discussion

The results presented in the preceding evaluations consistently show the tight connections between decisions at the level of reference, syntax and word order. These interactions entail highly interdependent modelling steps: Although there is a direct error propagation effect from predicted verb transformation on RE accuracy, predicted syntax still leads to informative intermediate linearizations that improve the RE prediction. Our optimal generation architecture thus has a sequential set-up, where the first linearization step can be seen as an intermediate feedback that is revised in the final linearization. This connects to work in, e.g. (Hovy, 1988; Robin, 1993).

In Figure 3, we compare two system outputs for the last two sentences of the text in Figure 2. The output of the sequential system is severely incoherent and would probably be rejected by a human reader: In sentence (5a) the *victim* subject of an active verb is deleted, and the relation between the possessive and the embedded *victim* RE is not clear. In sentence (5b) the first conjunct realizes a pronominal *perp* RE and the second conjunct a nominal *perp* RE. The output of the revision-based system reads much more natural. This example shows that the extension of the REG problem to texts with more than one main referent (as in the GREC data set) yields interesting inter-sentential interactions that affect textual coherence.

We are aware of the fact that our automatic eval-

uation might only partially render certain effects, especially with respect to textual coherence. It is likely that the BLEU scores do not capture the magnitude of the differences in text quality illustrated by the Examples (5-6). Ultimately, a human evaluation for this task is highly desirable. We leave this for future work since our integrated set-up rises a number of questions with respect to evaluation design. In a preliminary analysis, we noticed the problem that human readers find it difficult to judge discourse-level properties of a text like coherence or naturalness when the generation output is not perfectly grammatical or fluent at the sentence level.

## 6 Conclusion

We have presented a data-driven approach for investigating generation architectures that address discourse-level reference and sentence-level syntax and word order. The data set we created for our experiments basically integrates standards from previous research on REG and surface realization and extends the annotations to further types of implicit referents. Our results show that interactions between the different generation levels are best captured in a sequential, revision-based pipeline where the REG component has access to predictions from the syntax and the linearization module. These empirical findings obtained from experiments with generation architectures have clear connections to theoretical accounts of textual coherence.

## Acknowledgements

This work was supported by the Deutsche Forschungsgemeinschaft (German Research Foundation) in SFB 732 *Incremental Specification in Context*, project D2.

## References

- Douglas Edmund Appelt. 1982. *Planning natural language utterances to satisfy multiple goals*. Ph.D. thesis, Stanford, CA, USA.
- John Bateman and Michael Zock. 2003. Natural Language Generation. In Ruslan Mitkov, editor, *The Oxford Handbook of Computational Linguistics*. Oxford University Press.
- Anja Belz and Eric Kow. 2010. The GREC Challenges 2010: overview and evaluation results. In *Proc. of the 6th International Natural Language Generation Conference, INLG '10*, pages 219–229, Stroudsburg, PA, USA.
- Anja Belz and Sebastian Vargas. 2007. Generation of repeated references to discourse entities. In *Proc. of the 11th European Workshop on Natural Language Generation, ENLG '07*, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Anja Belz, Mike White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proc. of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 217–226, Nancy, France, September. Association for Computational Linguistics.
- Anja Belz. 2005. Statistical generation: Three methods compared and evaluated. In *Proc. of the 10th European Workshop on Natural Language Generation*, pages 15–23.
- Anja Belz. 2008. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Nat. Lang. Eng.*, 14(4):431–455, October.
- Bernd Bohnet, Leo Wanner, Simon Milles, and Alicia Burga. 2010. Broad coverage multilingual deep sentence generation with a stochastic multi-level realizer. In *Proc. of the 23rd International Conference on Computational Linguistics*, Beijing, China.
- Bernd Bohnet, Simon Mille, Benoît Favre, and Leo Wanner. 2011. <stumaba >: From deep representation to surface. In *Proc. of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 232–235, Nancy, France, September.
- Bernd Bohnet, Anders Björkelund, Jonas Kuhn, Wolfgang Seeker, and Sina Zarriess. 2012. Generating non-projective word order in statistical linearization. In *Proc. of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 928–939, Jeju Island, Korea, July.
- Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proc. of the 23rd International Conference on Computational Linguistics*, pages 89–97, Beijing, China, August.
- Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER Treebank. In *Proc. of the Workshop on Treebanks and Linguistic Theories*.
- Aoife Cahill and Arndt Riester. 2009. Incorporating Information Status into Generation Ranking. In *Proc. of the 47th Annual Meeting of the ACL*, pages 817–825, Suntec, Singapore, August.
- Lynne Cahill, Christy Doran, Roger Evans, Chris Melish, Daniel Paiva, Mike Reape, Donia Scott, and Neil Tipper. 1999. In search of a reference architecture for nlg systems. In *Proc. of the European Workshop on Natural Language Generation (EWNLG)*, pages 77–85.
- David L. Chen, Joohyun Kim, and Raymond J. Mooney. 2010. Training a multilingual sportscaster: Using perceptual context to learn language. *Journal of Artificial Intelligence Research*, 37:397–435.
- Robert Dale and Ehud Reiter. 1995. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263.
- Robert Dale. 1989. Cooking up referring expressions. In *Proc. of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 68–75, Vancouver, British Columbia, Canada, June.
- Laurence Danlos. 1984. Conceptual and linguistic decisions in generation. In *Proc. of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, pages 501–504, Stanford, California, USA, July.
- Koenraad De Smedt, Helmut Horacek, and Michael Zock. 1996. Architectures for natural language generation: Problems and perspectives. In *Trends In Natural Language Generation: An Artificial Intelligence Perspective*, pages 17–46. Springer-Verlag.
- Katja Filippova and Michael Strube. 2007. Generating constituent order in german clauses. In *Proc. of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic.
- Matthew Gerber and Joyce Chai. 2010. Beyond nombank: A study of implicit arguments for nominal predicates. In *Proc. of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1583–1592, Uppsala, Sweden, July.
- Barbara J. Grosz, Aravind Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

- Eduard H. Hovy. 1988. Planning coherent multisentential text. In *Proc. of the 26th Annual Meeting of the Association for Computational Linguistics*, pages 163–169, Buffalo, New York, USA, June.
- Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proc. of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 217–226.
- Irene Langkilde and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proc. of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 704–710, Montreal, Quebec, Canada, August. Association for Computational Linguistics.
- Tomasz Marciniak and Michael Strube. 2005. Beyond the pipeline: discrete optimization in nlp. In *Proc. of the 9th Conference on Computational Natural Language Learning, CONLL '05*, pages 136–143, Stroudsburg, PA, USA.
- Chris Mellish, Roger Evans, Lynne Cahill, Christy Doran, Daniel Paiva, Mike Reape, Donia Scott, and Neil Tipper. 2000. A representation for complex and evolving data dependencies in generation. In *Proc. of the 6th Conference on Applied Natural Language Processing*, pages 119–126, Seattle, Washington, USA, April.
- Marie Meteer. 1991. Bridging the generation gap between text planning and linguistic realization. In *Computational Intelligence*, volume 7 (4).
- Adwait Ratnaparkhi. 2000. Trainable methods for surface natural language generation. In *Proc. of the 1st North American chapter of the Association for Computational Linguistics conference, NAACL 2000*, pages 194–201, Stroudsburg, PA, USA.
- Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Nat. Lang. Eng.*, 3(1):57–87, March.
- Ehud Reiter. 1994. Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible? pages 163–170.
- Eric K. Ringger, Michael Gamon, Robert C. Moore, David Rojas, Martine Smets, and Simon Corston-Oliver. 2004. Linguistically Informed Statistical Models of Constituent Structure for Ordering in Sentence Realization. In *Proc. of the 2004 International Conference on Computational Linguistics*, Geneva, Switzerland.
- Jacques Robin. 1993. A revision-based generation architecture for reporting facts in their historical context. In *New Concepts in Natural Language Generation: Planning, Realization and Systems*. Frances Pinter, London and, pages 238–265. Pinter Publishers.
- Michael Roth and Anette Frank. 2012. Aligning predicate argument structures in monolingual comparable texts: A new corpus for a new task. In *Proc. of the 1st Joint Conference on Lexical and Computational Semantics (\*SEM)*, Montreal, Canada.
- Robert Rubinoff. 1992. Integrating text planning and linguistic choice by annotating linguistic structures. In Robert Dale, Eduard H. Hovy, Dietmar Rösner, and Oliviero Stock, editors, *NLG*, volume 587 of *Lecture Notes in Computer Science*, pages 45–56. Springer.
- Josef Ruppenhofer, Caroline Sporleder, Roser Morante, Collin Baker, and Martha Palmer. 2010. Semeval-2010 task 10: Linking events and their participants in discourse. In *Proc. of the 5th International Workshop on Semantic Evaluation*, pages 45–50, Uppsala, Sweden, July.
- Wolfgang Seeker and Jonas Kuhn. 2012. Making Ellipses Explicit in Dependency Conversion for a German Treebank. In *Proc. of the 8th conference on International Language Resources and Evaluation*, Istanbul, Turkey, May.
- Advaith Siddharthan and Ann Copestake. 2004. Generating referring expressions in open domains. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 407–414, Barcelona, Spain, July.
- Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for nlp-assisted text annotation. In *Proc. of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France, April.
- Leo Wanner, Simon Mille, and Bernd Bohnet. 2012. Towards a surface realization-oriented corpus annotation. In *Proc. of the 7th International Natural Language Generation Conference*, pages 22–30, Utica, IL, May.
- Leo Wanner. 1994. Building another bridge over the generation gap. In *Proc. of the 7th International Workshop on Natural Language Generation, INLG '94*, pages 137–144, Stroudsburg, PA, USA.
- Sina Zarriß, Aoife Cahill, and Jonas Kuhn. 2011. Underspecifying and predicting voice for surface realisation ranking. In *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1007–1017, Portland, Oregon, USA, June.
- Sina Zarriß, Aoife Cahill, and Jonas Kuhn. 2012. To what extent does sentence-internal realisation reflect discourse context? a study on word order. In *Proc. of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 767–776, Avignon, France, April.

# Named Entity Recognition using Cross-lingual Resources: Arabic as an Example

Kareem Darwish

Qatar Computing Research Institute

Doha, Qatar

kdarwish@qf.org.qa

## Abstract

Some languages lack large knowledge bases and good discriminative features for Name Entity Recognition (NER) that can generalize to previously unseen named entities. One such language is Arabic, which: a) lacks a capitalization feature; and b) has relatively small knowledge bases, such as Wikipedia. In this work we address both problems by incorporating cross-lingual features and knowledge bases from English using cross-lingual links. We show that such features have a dramatic positive effect on recall. We show the effectiveness of cross-lingual features and resources on a standard dataset as well as on two new test sets that cover both news and microblogs. On the standard dataset, we achieved a 4.1% relative improvement in F-measure over the best reported result in the literature. The features led to improvements of 17.1% and 20.5% on the new news and microblogs test sets respectively.

## 1 Introduction

Named Entity Recognition (NER) is essential for a variety of Natural Language Processing (NLP) applications such as information extraction. There has been a fair amount of work on NER for a variety of languages including Arabic. To train an NER system, some of the following feature types are typically used (Benajiba and Rosso, 2008; Nadeau and Sekine, 2009):

- Orthographic features: These features include capitalization, punctuation, existence of digits, etc. One of the most effective orthographic features is capitalization in English, which helps NER to generalize to new text of different genres. However, capitalization is not very useful in some languages such as German, and nonexistent in other languages such

as Arabic. Further, even in English social media, capitalization may be inconsistent.

- Contextual features: Certain words are indicative of the existence of named entities. For example, the word “said” is often preceded by a named entity of type “person” or “organization”. Sequence labeling algorithms (ex. Conditional Random Fields (CRF)) can often identify such indicative words.

- Character-level features: These features typically include the leading and trailing letters of words. In some languages, these letters could be prefixes and suffixes. Such features can be indicative or counter-indicative of the existence of named entities. For example, a word ending with “ing” is typically not a named entity, while a word ending in “berg” is often a named entity.

- Part-of-speech (POS) tags and morphological features: POS tags indicate (or counter-indicate) the possible presence of a named entity at word level or at word sequence level. Morphological features can mostly indicate the absence of named entities. For example, Arabic allows the attachment of pronouns to nouns and verbs. However, pronouns are rarely ever attached to named entities.

- Gazetteers: This feature checks the presence of a word or a sequence of words in large lists of named entities. If gazetteers are small, then they would have low coverage, and if they are very large then their entries may be ambiguous. For example, “syntax” may refer to sentence construction or the music band “Syntax”.

Typically, a subset of these features are available for different languages. For example, morphological, contextual, and character-level features have been shown to be effective for Arabic NER (Benajiba and Rosso, 2008). However, Arabic lacks indicative orthographic features that generalize to previously unseen named entities. Also, although some

of the Arabic gazetteers that were used for NER were small (Benajiba and Rosso, 2008), there has been efforts to build larger Arabic gazetteers (Attia et al., 2010). Since training and test parts of standard datasets for Arabic NER are drawn from the same genre in relatively close temporal proximity, a named entity recognizer that simply memorizes named entities in the training set generally performs well on such test sets. Thus, the results that are reported in the literature are generally high (Abdul-Hamid and Darwish, 2010; Benajiba et al., 2008). We illustrate the limited capacity of existing recognizers to generalize to previously unseen named entities using two new test sets that include microblogs as well as news texts that cover local and international politics, economics, health, sports, entertainment, and science. As we will show later, recall is well below 50% for all named entity types on the new test sets.

To address this problem, we introduce the use of cross-lingual links between a disadvantaged language, Arabic, and a language with good discriminative features and large resources, English, to improve Arabic NER. We exploit English’s orthographic features, particularly capitalization, as well as Arabic and English Wikipedias, including existing annotations from large knowledge sources such as DBpedia. We also show how to use transliteration mining to improve NER, even when neither language has a capitalization (or similar) feature. The intuition is that if the translation of a word is in fact a transliteration, then the word is likely a named entity. Cross-lingual links are obtained using Wikipedia cross-language links and a large Machine Translation (MT) phrase table that is true cased, where word casing is preserved during training. We show the effectiveness of these new features on a standard dataset as well as two new test sets. The contributions of this paper are as follows:

- Using cross-lingual links to exploit orthographic features in other languages.
- Employing transliteration mining to improve NER.
- Using cross-lingual links to exploit a large knowledge base, namely English DBpedia, to benefit NER.
- Introducing two new NER test sets for Arabic that include recent news as well as microblogs. We plan to release these test sets.

- Improving over the best reported results in the literature by 4.1% (Abdul-Hamid and Darwish, 2010) by strictly adding cross-lingual features. We also show improvements of 17.1% and 20.5% on the new test sets.

The remainder of the paper is organized as follows: Section 2 provides related work; Section 3 describes the baseline system; Section 4 introduces the cross-lingual features and reports on their effectiveness; and Section 5 concludes the paper.

## 2 Related Work

### 2.1 Using cross-lingual Features

For many NLP tasks, some languages may have significantly more training data, better knowledge resources, or more discriminating features than other languages. If cross-lingual resources are available, such as parallel data, increased training data, better resources, or superior features can be used to improve the processing (ex. tagging) for other languages (Ganchev et al., 2009; Shi et al., 2010; Yarowsky and Ngai, 2001). Some work has attempted to use bilingual features in NER. Burkett et al. (2010) used bilingual text to improve monolingual models including NER models for German, which lacks a good capitalization feature. They did so by training a bilingual model and then generating more training data from unlabeled parallel data. They showed significant improvement in German NER effectiveness, particularly for recall. In our work, there is no need for tagged text that has a parallel equivalent in another language. Benajiba et al. (2008) used an Arabic English dictionary from MADA, an Arabic analyzer, to indicate if a word is capitalized in English or not. They reported that it was the second most discriminating feature that they used. However, there seems to be room for improvement because: (1) MADA’s dictionary is relatively small and would have low coverage; and (2) the use of such a binary feature is problematic, because Arabic names are often common Arabic words and hence a word may be translated as a named entity and as a common word. To overcome these two problems, we use cross-lingual features to improve NER using large bilingual resources, and we incorporate confidences to avoid having a binary feature. Richman and Schone (2008) used English linguis-

tic tools and cross language links in Wikipedia to automatically annotate text in different languages. Transliteration Mining (TM) has been used to enrich MT phrase tables or to improve cross language search (Udupa et al., 2009). Conversely, people have used NER to determine if a word is to be transliterated or not (Hermjakob et al., 2008). However, we are not aware of any prior work on using TM to determine if a sequence is a NE. Further, we are not aware of prior work on using TM (or transliteration in general) as a cross lingual feature in any annotation task. In our work, we use state-of-the-art TM as described by El-Kahki et al. (2011)

## 2.2 Arabic NER

Much work has been done on NER with multiple public evaluation forums. Nadeau and Sekine (Nadeau and Sekine, 2009) surveyed lots of work on NER for a variety of languages. Significant work has been conducted by Benajiba and colleagues on Arabic NER (Benajiba and Rosso, 2008; Benajiba et al., 2008; Benajiba and Rosso, 2007; Benajiba et al., 2007). Benajiba et al. (2007) used a maximum entropy classifier trained on a feature set that includes the use of gazetteers and a stop-word list, appearance of a NE in the training set, leading and trailing word bigrams, and the tag of the previous word. They reported 80%, 37%, and 47% F-measure for locations, organizations, and persons respectively on the ANERCORP dataset that they created and publicly released. Benajiba and Rosso (2007) improved their system by incorporating POS tags to improve NE boundary detection. They reported 87%, 46%, and 52% F-measure for locations, organizations, and persons respectively. Benajiba and Rosso (2008) used CRF sequence labeling and incorporated many language specific features, namely POS tagging, base-phrase chunking, Arabic tokenization, and adjectives indicating nationality. They reported that tokenization generally improved recall. Using POS tagging generally improved recall at the expense of precision, leading to overall improvements in F-measure. Using all their suggested features, they reported 90%, 66%, and 73% F-measure for location, organization, and persons respectively. In Benajiba et al. (2008), they examined the same feature set on the Automatic Content Extraction (ACE) datasets using CRF

sequence labeling and a Support Vector Machine (SVM) classifier. They did not report per category F-measure, but they reported overall 81%, 75%, and 78% macro-average F-measure for broadcast news and newswire on the ACE 2003, 2004, and 2005 datasets respectively. Huang (2005) used an HMM-based NE recognizer for Arabic and reported 77% F-measure on the ACE 2003 dataset. Farber et al. (2008) used POS tags obtained from an Arabic tagger to enhance NER. They reported 70% F-measure on the ACE 2005 dataset. Shaalan and Raza (2007) reported on a rule-based system that uses hand crafted grammars and regular expressions in conjunction with gazetteers. They reported upwards of 93% F-measure, but they conducted their experiments on non-standard datasets, making comparison difficult. Abdul-Hamid and Darwish (2010) used a simplified feature set that relied primarily on character level features, namely leading and trailing letters in a word. They also experimented with a variety of phrase level features with little success. They reported an F-measure of 76% and 81% for the ACE2005 and the ANERCORP datasets respectively. We used their simplified features in our baseline system. The different experiments reported in the literature may not have been done on the same training/test splits. Thus, the results may not be completely comparable. Mohit et al. (2012) performed NER on a different genre from news, namely Arabic Wikipedia articles, and reported recall values as low as 35.6%. They used self training and recall oriented classification to improve recall, typically at the expense of precision. McNamee and Mayfield (2002) and Mayfield et al. (2003) used thousands of language independent features such as character n-grams, capitalization, word length, and position in a sentence, along with language dependent features such as POS tags and BP chunking. The use of CRF sequence labeling for NER has shown success (McCallum and Li, 2003; Nadeau and Sekine, 2009; Benajiba and Rosso, 2008).

## 3 Baseline Arabic NER System

For the baseline system, we used the CRF++<sup>1</sup> implementation of CRF sequence labeling with default parameters. We opted to reimplement the most suc-

<sup>1</sup><http://code.google.com/p/crfpp/>

cessful features that were reported by Benajiba et al. (2008) and Abdul-Hamid and Darwish (2010), namely the leading and trailing 1, 2, 3, and 4 letters in a word; whether a word appears in the gazetteer that was created by Benajiba et al. (2008), which is publicly available, but is rather small (less than 5,000 entries); and the stemmed form of words (after removing coordinating conjunctions, prepositions, and determiners using a rule-based stemmer akin to (Larkey et al., 2002)). As mentioned earlier, the leading and trailing letters in a word may indicate or counter-indicate the presence of named entities. Stemming is important due to the morphological complexity of Arabic. We used the previous and the next words in their raw and stemmed forms as features. For training and testing, we used the ANERCORP dataset (Benajiba and Rosso, 2007). The dataset has approximately 150k tokens and we used the 80/20 training/test splits of Abdul-Hamid and Darwish (2010), who graciously provided us with their splits of the collection and they achieved the best reported results on the dataset. We will refer to their results, which are provided in Table 1, as “baseline-lit”. Table 2 (a) shows our results on the ANERCORP dataset. Our results were slightly lower than their results (Abdul-Hamid and Darwish, 2010). It is noteworthy that 69% of the named entities in the test part were seen during training.

We also created two new test sets. The first test set is composed of news snippets from the RSS feed of the Arabic (Egypt) version of news.google.com from Oct. 6, 2012. The RSS feed contains the headline and the first 50-100 words in the news articles. The set has news from over a dozen different news sources and covers international and local news, politics, financial news, health, sports, entertainment, and technology. This set contains roughly 15k tokens. The second set contains a set of 1,423 tweets that were randomly selected from tweets authored between November 23, 2011 and November 27, 2011. We scraped tweets from Twitter using the query “lang:ar” (language=Arabic). This set contains approximately 26k tokens. The test sets will be henceforth be referred to as the NEWS and TWEETS sets respectively. They were annotated by one person, a native Arabic speaker, using the Linguistics Data Consortium tagging guidelines. Table 2 (b) and (c) report on the results for the baseline

system on both test sets. The results on the NEWS test are substantially lower than those for ANERCORP. It is worth noting that only 27% of the named entities in the NEWS test set were observed in the training set (compared to 69% for ANERCORP). As Table 3 shows for the ANERCORP dataset, using only the tokens as features, where the labeler mainly memorizes previously seen named entities, yields higher results than the baseline results for the NEWS dataset (Table 2 (b)). The results on the TWEETS test are very poor, with 24% of the named entities in the test set appearing in the training set.

ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	93	83	88
ORG	84	64	73
PERS	90	75	82
Overall	89	74	81

Table 1: “Baseline-lit” Results from (Abdul-Hamid and Darwish, 2010)

(a) ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	93.6	83.3	88.1
ORG	83.8	61.2	70.8
PERS	84.3	64.4	73.0
Overall	88.9	72.5	79.9
(b) NEWS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	84.1	53.2	65.1
ORG	73.2	23.2	35.2
PERS	74.8	47.1	57.8
Overall	78.0	41.9	54.6
(c) TWEETS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	79.9	27.1	40.4
ORG	44.4	9.1	15.1
PERS	45.7	27.8	34.5
Overall	58.0	23.1	33.1

Table 2: Baseline Results for the three test sets

ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	95.3	62.7	75.6
ORG	86.3	44.7	58.9
PERS	85.4	36.4	51.0
Overall	91.0	50.0	64.5

Table 3: Results of using only tokens as features on ANERCORP

## 4 Cross-lingual Features

We experimented with three different cross-lingual features that used Arabic and English Wikipedia cross-language links and a true-cased phrase table that was generated using Moses (Koehn et al., 2007). True-casing preserves case information during training. We used the Arabic Wikipedia snapshot from September 28, 2012. The snapshot has 348,873 titles including redirects, which are alternative names to articles. Of these articles, 254,145 have cross-lingual links to English Wikipedia. We used DBpedia 3.8 which includes 6,157,591 entries of Wikipedia titles and their “types”, such as “person”, “plant”, or “device”, where a title can have multiple types. The phrase table was trained on a set of 3.69 million parallel sentences containing 123.4 million English tokens. The sentences were drawn from the UN parallel data along with a variety of parallel news data from LDC and the GALE project. The Arabic side was stemmed (by removing just prefixes) using the Stanford word segmenter (Green and DeNero, 2012).

### 4.1 Cross-lingual Capitalization

As we mentioned earlier, Arabic lacks capitalization and Arabic names are often common Arabic words. For example, the Arabic name “Hasan” means good. To capture cross-lingual capitalization, we used the aforementioned true-cased phrase table at word and phrase levels as follows:

**Input:** True-cased phrase table  $PT$ , sentence  $S$  containing  $n$  words  $w_{0..n}$ , max sequence length  $l$ , translations  $T_{1..k..m}$  of  $w_{i..j}$

```

for  $i = 0 \rightarrow n$  do
   $j = \min(i + l - 1, n)$ 
  if  $PT$  contains  $w_{i..j}$  &  $\exists T_k$   $isCaps$  then
     $weight(w_{i..j}) = \frac{\sum_{T_k.isCaps} P(T_k)}{\sum_{T_k.isCaps} P(T_k) + \sum_{T_k.notCaps} P(T_k)}$ 
    round  $weight(w_{i..j})$  to first significant figure
    set tag of  $w_i = B-CAPS-weight$ 
    set tag for words  $w_{i+1..j} = I-CAPS-weight$ 
  else
    if  $j > i$  then
       $j--$ 
    else
      tag of  $w_i = null$ 
    end if
  end if
end for

```

Where:  $PT$  was the aforementioned phrase table;  $l = 4$ ;  $P(T_k)$  equaled to the product of  $p(source|target)$  and  $p(target|source)$  for a word sequence;  $isCaps$  and  $notCaps$  were whether the

translation was capitalized or not respectively; and the weights were binned because CRF++ only takes nominal features. In essence we tried every subsequence of  $S$  of length  $l$  or less to see if the translation was capitalized. A subsequence can be 1 word long. We tried longer sequences first. To determine if the corresponding phrase was capitalized ( $isCaps$ ), all non-function words on the English side needed to be capitalized. As an example, the phrase المحيط الهادي (meaning “Pacific Ocean”) was translated to a capitalized phrase 36.7% of the time. Thus, the word المحيط was assigned B-CAPS-0.4 and الهادي was assigned I-CAPS-0.4. Using weights avoids using capitalization as a binary feature.

Table 4 reports on the results of the baseline system with the capitalization feature on the three datasets. In comparing baseline results in Table 2 and cross-lingual capitalization results in Table 4, recall consistently increased for all datasets, particularly for “persons” and “locations”. For the different test sets, recall increased by 3.1 to 6.1 points (absolute) or by 8.4% to 13.6% (relative). This led to an overall improvement in F-measure of 1.8 to 3.4 points (absolute) or 4.2% to 5.7% (relative). Precision dropped overall on the ANERCORP dataset and dropped substantially for the NEWS and TWEETS test sets.

(a) ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	92.0/-1.6/-1.7	86.8/3.5/4.2	89.3/1.2/1.4
ORG	82.8/-1.1/-1.3	63.9/2.7/4.4	72.1/1.4/1.9
PERS	86.0/1.7/2.0	75.4/11.0/17.1	80.3/7.3/10.1
Overall	88.4/-0.4/-0.5	78.6/6.1/8.4	83.2/3.4/4.2
(b) NEWS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	82.1/-2.0/-2.4	59.0/5.8/11.0	68.7/3.5/5.4
ORG	68.4/-4.9/-6.6	23.2/0.0/0.0	34.6/-0.6/-1.7
PERS	70.7/-4.0/-5.4	55.6/8.4/17.9	62.2/4.4/7.6
Overall	74.5/-3.5/-4.5	47.0/5.1/12.2	57.7/3.1/5.7
(c) TWEETS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	76.9/-3.0/-3.7	27.9/0.9/3.2	41.0/0.5/1.4
ORG	44.4/0.0/0.0	10.4/1.3/14.3	16.8/1.8/11.6
PERS	40.0/-5.7/-12.5	35.0/7.3/26.2	37.3/2.8/8.1
Overall	51.8/-6.2/-10.7	26.3/3.1/13.6	34.9/1.8/5.4

Table 4: Results with cross-lingual capitalization with /absolute/relative differences compared to baseline

## 4.2 Transliteration Mining

An alternative to capitalization can be transliteration mining. The intuition is that named entities are often transliterated, particularly the names of locations and persons. This feature is helpful if cross-lingual resources do not have capitalization information, or if the “helper” language to be consulted does not have a useful capitalization feature. We performed transliteration mining (aka cognate matching) at word level for each Arabic word against all its possible translations in the phrase table. We used a transliteration miner akin to that of El-Kahki et al. (2011) that was trained using 3,452 parallel Arabic-English transliteration pairs. We aligned the word-pairs at character level using GIZA++ and the phrase extractor and scorer from the Moses machine translation package (Koehn et al., 2007). The alignment produced mappings between English letters sequences and Arabic letter sequences with associated mapping probabilities. Given an Arabic word, we produced all its possible segmentations along with their associated mappings into English letters. We retained valid target sequences that produced translations in the phrase table.

Again we used a weight similar to the one for cross-lingual capitalization and we rounded the values of the ratio the significant figure. The weights were computed as:

$$\frac{\sum_{T_k \text{ is Transliteration}} P(T_k)}{\sum_{T_k \text{ is Transliteration}} P(T_k) + \sum_{T_k \text{ not Transliteration}} P(T_k)} \quad (1)$$

where  $P(T_k)$  is probability of the  $k^{\text{th}}$  translation of a word in the phrase table.

If a word was not found in the phrase table, the feature value was assigned null. For example, if the translations of the word حسن are “Hasan”, “Hassan”, and “good”, where the first two are transliterations and the last not, then the weight of the word would be:

$$\frac{P(\text{Hasan}|\text{حسن}) + P(\text{Hassan}|\text{حسن})}{P(\text{Hasan}|\text{حسن}) + P(\text{Hassan}|\text{حسن}) + P(\text{good}|\text{حسن})} \quad (2)$$

In our experiments, the weight of حسن was equal to 0.5 (after rounding). Table 5 reports on the results using the baseline system with the transliteration mining feature. Like the capitalization fea-

ture, transliteration mining slightly lowered precision – except for the TWEETS test set where the drop in precision was significant – and positively increased recall, leading to an overall improvement in F-measure for all test sets. Overall, F-measure improved by 1.9%, 3.7%, and 3.9% (relative) compared to the baseline for the ANERCORP, NEWS, and TWEETS test sets respectively. The similarity of results between using transliteration mining and word-level cross-lingual capitalization suggests that perhaps they can serve as surrogates.

## 4.3 Using DBpedia

DBpedia<sup>2</sup> is a large collaboratively-built knowledge base in which structured information is extracted from Wikipedia (Bizer et al., 2009). DBpedia 3.8, the release we used in this paper, contains 6,157,591 Wikipedia titles belonging to 296 types. Types vary in granularity with each Wikipedia title having one or more type. For example, NASA is assigned the following types: Agent, Organization, and GovernmentAgency. In all, DBpedia includes the names of 764k persons, 573k locations, and 192k organizations. Of the Arabic Wikipedia titles, 254,145 have Wikipedia cross-lingual links to English Wikipedia, and of those English Wikipedia titles, 185,531 have entries in DBpedia. Since Wikipedia titles may have multiple DBpedia types, we opted to keep the most popular type (by count of how many Wikipedia titles are assigned a particular type) for each title, and we disregarded the rest. We also chose not to use the “Agent” and “Work” types because they were highly ambiguous. We found word sequences in the manner described in the pseudocode for cross-lingual capitalization. For translation, we generated two features using two translation resources, namely the aforementioned phrase table and Arabic-English Wikipedia cross-lingual links. When using the phrase table, we used the most likely translation into English that matches an entry in DBpedia provided that the product of  $p(\text{source}|\text{target})$  and  $p(\text{target}|\text{source})$  of translation was above  $10^{-5}$ . We chose the threshold qualitatively using offline experiments. When using Arabic-English Wikipedia cross-lingual links, if an entry was found in the Arabic Wikipedia, we performed a look up in DB-

<sup>2</sup><http://dbpedia.org>

pedia using the English Wikipedia title that corresponds to the Arabic Wikipedia title. We used Arabic Wikipedia page-redirects to improve coverage. For both features (using the two translation methods), for an Arabic word sequence corresponding to the DBpedia entry, the first word in the sequence was assigned the feature “B-” plus the DBpedia type and subsequent words were assigned the feature “I-” plus the DBpedia type. For example, for حزب الله (meaning “Hezbollah”), the words حزب and الله were assigned “B-Organization” and “I-Organization” respectively. For all other words, the feature was assigned “null”. Using the phrase table for translation likely yielded improved coverage over using Wikipedia cross-lingual links. However, Wikipedia cross-lingual links likely led to higher quality translations, because they were manually curated. Table 6 reports on the results of using the baseline system with the two DBpedia features. Using DBpedia consistently improved precision and recall for named entity types on all test sets, except for a small drop in precision for locations on the ANERCORP dataset and for locations and persons on the TWEETS test set. For the different test sets, improvements in recall ranged between 4.4 and 7.5 points (absolute) or 6.5% and 19.1% (relative). Precision improved by 0.9 and 5.5 points (absolute) or 1.0% and 7.1% (relative) for the ANERCORP and NEWS test sets respectively. Overall improvement in F-measure ranged between 3.2 and 7.6 points (absolute) or 4.1% and 13.9% (relative).

#### 4.4 Putting it All Together

Table 7 reports on the results of using all aforementioned cross-lingual features together. Figures 1, 2, and 3 compare the results of the different setups. As the results show, the impact of cross-lingual features on recall were much more pronounced on the NEWS and TWEETS test sets – compared to the ANERCORP dataset. Further, the recall values for the ANERCORP dataset in the baseline experiments were much higher than those for the two other test sets. This confirms our suspicion that the reported values in the literature on the standard datasets are unrealistically high due to the similarity between the training and test sets. Hence, these high effectiveness results may not generalize to other test sets. Of all the cross-

(a) ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	92.9/-0.7/-0.7	83.5/0.2/0.3	88.0/-0.2/-0.2
ORG	82.9/-0.9/-1.0	61.8/0.6/1.0	70.9/0.1/0.1
PERS	84.5/0.3/0.3	71.9/7.5/11.7	77.7/4.7/6.5
Overall	88.3/-0.5/-0.6	75.5/2.9/4.1	81.4/1.5/1.9
(b) NEWS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	84.9/0.7/0.9	53.6/0.5/0.9	65.7/0.6/0.9
ORG	67.2/-6.1/-8.3	22.9/-0.3/-1.1	34.2/-1.0/-2.9
PERS	72.8/-1.9/-2.6	55.0/7.8/16.7	62.7/4.8/8.4
Overall	75.9/-2.1/-2.6	45.0/3.1/7.4	56.6/2.0/3.7
(c) TWEETS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	79.1/-0.8/-1.0	27.1/0.0/0.0	40.3/-0.1/-0.3
ORG	41.8/-2.7/-6.0	9.1/0.0/0.0	14.9/-0.2/-1.1
PERS	40.0/-5.7/-12.5	35.5/7.7/27.8	37.6/3.1/8.8
Overall	51.7/-6.3/-10.9	25.8/2.6/11.3	34.4/1.3/3.9

Table 5: Results with transliteration mining with /absolute/relative differences compared to baseline

(a) ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	92.7/-0.9/-0.9	87.1/3.9/4.6	89.9/1.7/1.9
ORG	84.6/0.8/0.9	66.6/5.3/8.7	74.5/3.7/5.3
PERS	87.8/3.6/4.2	69.9/5.5/8.6	77.8/4.8/6.6
Overall	89.8/0.9/1.0	77.2/4.7/6.5	83.0/3.2/4.0
(b) NEWS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	87.8/3.6/4.3	61.8/8.6/16.2	72.5/7.4/11.3
ORG	76.1/2.9/3.9	30.2/7.0/30.1	43.2/8.0/22.7
PERS	83.2/8.5/11.3	54.2/7.1/15.0	65.7/7.8/13.6
Overall	83.5/5.5/7.1	49.5/7.5/18.0	62.2/7.6/13.9
(c) TWEETS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	77.4/-2.5/-3.1	30.5/3.5/12.9	43.8/3.4/8.4
ORG	57.0/12.5/28.2	15.9/6.8/75.1	24.8/9.8/64.9
PERS	40.8/-4.9/-10.6	31.7/4.0/14.3	35.7/1.2/3.4
Overall	55.3/-2.6/-4.5	27.5/4.4/19.1	36.8/3.7/11.2

Table 6: Results using DBpedia with /absolute/relative differences compared to baseline

lingual features that we experimented with, the use of DBpedia led to improvements in both precision and recall (except for precision on the TWEETS test set). Other cross-lingual features yielded overall improvements in F-measure, mostly due to gains in recall, typically at the expense of precision. Overall, F-measure improved by 5.5%, 17.1%, and 20.5% (relative) compared to the baseline for the ANERCORP, NEWS, and TWEETS test sets respectively. For the ANERCORP test set, our results improved over the baseline-lit results (Abdul-Hamid and Darwish, 2010) by 4.1% (relative).

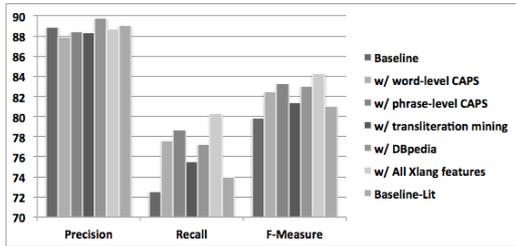


Figure 1: ANERCORP Dataset Results

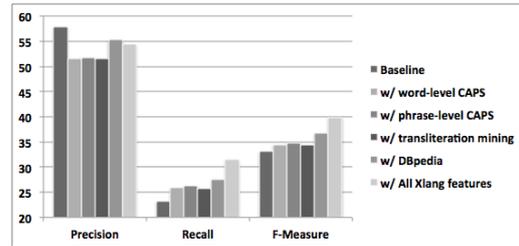


Figure 3: TWEETS Test Set Results

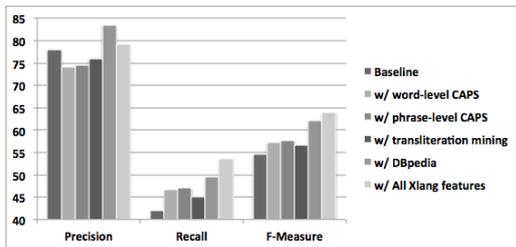


Figure 2: NEWS Test Set Results

When using all the features together, one notable result is that precision dropped significantly for the TWEETS test sets. We examined the output for the TWEETS test set and here are some of the factors that affected precision:

- the presence of words that would typically be named entities in news but would generally be regular words in tweets. For example, the Arabic word “Mubarak” is most likely the name of the former Egyptian president in the context of news, but it would most likely mean “blessed”, which is common in expressions of congratulations, in tweets.
- the use of dialectic words that may have transliterations or a named entity as the most likely translation into English. For example, the word شي is typically the dialectic version of the Arabic word شيء, meaning something. However, since the MT system that we used was trained on modern standard Arabic, the dialectic word would not appear in training and would typically be translated/transliterated to the name “Che” (as in Che Guevara).
- Since tweets are restricted in length, authors frequently use shortened versions of named entities. For example, tweets would mostly have “Morsi” instead of “Mohamed Morsi” and without trigger words such as “Dr.” or “president”. The full version of a name and trigger words are more com-

(a) ANERCORP Dataset			
	Precision	Recall	$F_{\beta=1}$
LOC	92.3/-1.3/-1.4	87.8/4.6/5.5	90.0/1.9/2.1
ORG	81.4/-2.4/-2.9	66.0/4.7/7.7	72.9/2.1/3.0
PERS	87.0/2.8/3.3	77.7/13.3/20.7	82.1/9.1/12.5
Overall	88.7/-0.2/-0.2	80.3/7.8/10.7	84.3/4.4/5.5
(b) NEWS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	85.1/1.0/1.2	64.1/11.0/20.6	73.1/8.0/12.3
ORG	73.8/0.5/0.7	29.4/6.2/26.9	42.1/6.8/19.4
PERS	76.8/2.0/2.7	63.4/16.3/34.5	69.5/11.7/20.2
Overall	79.2/1.2/1.6	53.6/11.6/27.7	63.9/9.4/17.1
(c) TWEETS Test Set			
	Precision	Recall	$F_{\beta=1}$
LOC	81.4/1.5/1.8	33.5/6.5/23.9	47.5/7.1/17.4
ORG	52.1/7.6/17.2	16.2/7.1/78.6	24.7/9.6/64.1
PERS	40.5/-5.2/-11.4	39.2/11.5/41.3	39.8/5.3/15.4
Overall	54.4/-3.6/-6.2	31.4/8.3/35.9	39.9/6.8/20.5

Table 7: Results using all the cross-lingual features with /absolute/relative differences compared to baseline

mon in news. This same problem was present in the NEWS test set, because it was constructed from an RSS feed, and headlines, which are typically compact, had a higher representation in the test collection. We observed the same phenomenon for organization names. For example, “the Real” refers to “Real Madrid”. Nicknames are also prevalent. For example, “the Land of the two Sanctuaries” refers to “Saudi Arabia”.

We believe that this problem can be overcome by introducing new training data that include tweets (or other social text) and performing domain adaptation. New training data would help: identify words and expressions that are common in conversations, account for common dialectic words, and learn a better word transition model. Further, gazetteers that cover shortened versions of names could be helpful as well.

## 5 Conclusion

In this paper, we presented different cross-lingual features that can make use of linguistic properties and knowledge bases of other languages for NER. For translation, we used an MT phrase table and Wikipedia cross-lingual links. We used English as the “helper” language and we exploited the English capitalization feature and an English knowledge base, DBpedia. If the helper language did not have capitalization, then transliteration mining could provide some of the benefit of capitalization. Transliteration mining requires limited amounts of training examples. We believe that the proposed cross-lingual features can be used to help NER for other languages, particularly languages that lack good features that generalize well. For Arabic NER, the new features yielded an improvement of 5.5% over a strong baseline system on a standard dataset, with 10.7% gain in recall and negligible change in precision. We tested on a new news test set, NEWS, which has recent news articles (the same genre as the standard dataset), and indeed NER effectiveness was much lower. For the new NEWS test set, cross-lingual features led to a small increase in precision (1.6%) and a very large improvement in recall (27.7%). This led to a 17.1% improvement in overall F-measure. We also tested NER on the TWEETS test set, where we observed substantial improvements in recall (35.9%). However, precision dropped by 6.2% for the reasons we mentioned earlier. For future work, it would be interesting to apply cross-lingual features to other language pairs and to make use of joint cross-lingual models. Further, we also plan to investigate Arabic NER on social media, particularly microblogs.

## References

- A. Abdul-Hamid and K. Darwish. 2010. Simplified Feature Set for Arabic Named Entity Recognition. Proceedings of the 2010 Named Entities Workshop, ACL 2010, pages 110115.
- Mohammed Attia, Antonio Toral, Lamia Tounsi, Monica Monachini, and Josef van Genabith. 2010. An automatically built named entity lexicon for Arabic. In: LREC 2010 - 7th conference on International Language Resources and Evaluation, 17-23 May 2010, Valletta, Malta.
- Y. Benajiba, M. Diab, and P. Rosso. 2008. Arabic Named Entity Recognition using Optimized Feature Sets. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 284293, Honolulu, October 2008.
- Y. Benajiba and P. Rosso. 2008. Arabic Named Entity Recognition using Conditional Random Fields. In Proc. of Workshop on HLT & NLP within the Arabic World, LREC08.
- Y. Benajiba, P. Rosso and J. M. Benedi. 2007. ANERsys: An Arabic Named Entity Recognition system based on Maximum Entropy. In Proc. of CILCling-2007, Springer-Verlag, LNCS(4394), pp.143-153
- Y. Benajiba and P. Rosso. 2007. ANERsys 2.0: Conquering the NER task for the Arabic language by combining the Maximum Entropy with POS-tag information. In Proc. of Workshop on Natural Language-Independent Engineering, IICAI-2007.
- Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sren Auer, Christian Becker, Richard Cyganiak, Sebastian Hellmann. 2009. DBpedia A Crystallization Point for the Web of Data. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Issue 7, Pages 154165, 2009.
- D. Burkett, S. Petrov, J. Blitzer, D. Klein. 2010. Learning Better Monolingual Models with Unannotated Bilingual Text. Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pages 46–54.
- A. El Kahki, K. Darwish, A. Saad El Din, M. Abd El-Wahab and A. Hefny. 2011. Improved Transliteration Mining Using Graph Reinforcement. EMNLP-2011.
- B. Farber, D. Freitag, N. Habash, and O. Rambow. 2008. Improving NER in Arabic Using a Morphological Tagger. In Proc. of LREC08.
- K. Ganchev, J. Gillenwater, and B. Taskar. 2009. Dependency grammar induction via bitext projection constraints. In ACL-2009.
- Spence Green and John DeNero. 2012. A Class-Based Agreement Model for Generating Accurately Inflected Translations. In ACL-2012.
- Ulf Hermjakob, Kevin Knight, and Hal Daum III. 2008. Name translation in statistical machine translation: Learning when to transliterate. ACL-08: HLT, Pages 389-397.
- F. Huang. 2005. Multilingual Named Entity Extraction and Translation from Text and Speech. Ph.D. Thesis. Pittsburgh: Carnegie Mellon University.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst, Moses: Open Source Toolkit

- for Statistical Machine Translation, Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic, June 2007.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, In Proc. of ICML, pp.282-289, 2001.
- Leah S. Larkey, Lisa Ballesteros, and Margaret E. Connell. 2002. Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis. SIGIR-2002.
- J. Mayfield, P. McNamee, and C. Piatko. 2003. Named Entity Recognition using Hundreds of Thousands of Features. HLT-NAACL 2003-Volume 4, 2003.
- A. McCallum and W. Li. 2003. Early Results for Named Entity Recognition with Conditional Random Fields, Features Induction and Web-Enhanced Lexicons. In Proc. Conference on Computational Natural Language Learning.
- P. McNamee and J. Mayfield. 2002. Entity extraction without language-specific. Proceedings of CoNLL, 2002
- Behrang Mohit, Nathan Schneider, Rishav Bhowmick, Kemal Oflazer, Noah A. Smith. 2012. Recall-oriented learning of named entities in Arabic Wikipedia. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012), pp. 162-173. 2012.
- D. Nadeau and S. Sekine. 2009. A Survey of Named Entity Recognition and Classification. Named Entities: Recognition, Classification and Use, ed. S. Sekine and E. Ranchhod, John Benjamins Publishing Company.
- Alexander E. Richman and Patrick Schone. 2008. Mining wiki resources for multilingual named entity recognition. Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2008.
- K. Shaalan and H. Raza. 2007. Person Name Entity Recognition for Arabic. Proceedings of the 5th Workshop on Important Unresolved Matters, pages 1724, Prague, Czech Republic, June 2007.
- L. Shi, R. Mihalcea, M. Tian. 2010. Cross Language Text Classification by Model Translation and Semi-supervised Learning. Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2010.
- Raghavendra Udupa, Anton Bakalov, and Abhijit Bhole. 2009. They Are Out There, If You Know Where to Look: Mining Transliterations of OOV Query Terms for Cross-Language Information Retrieval. Advances in Information Retrieval. Pages: 437-448.
- D. Yarowsky and G. Ngai. 2001. Inducing Multilingual POS Taggers and NP Bracketers via Robust Projection across Aligned Corpora. In NAACL-2001.

# Beam Search for Solving Substitution Ciphers

Malte Nuhn and Julian Schamper and Hermann Ney

Human Language Technology and Pattern Recognition

Computer Science Department, RWTH Aachen University, Aachen, Germany

<surname>@cs.rwth-aachen.de

## Abstract

In this paper we address the problem of solving substitution ciphers using a beam search approach. We present a conceptually consistent and easy to implement method that improves the current state of the art for decipherment of substitution ciphers and is able to use high order  $n$ -gram language models. We show experiments with 1:1 substitution ciphers in which the guaranteed optimal solution for 3-gram language models has 38.6% decipherment error, while our approach achieves 4.13% decipherment error in a fraction of time by using a 6-gram language model. We also apply our approach to the famous Zodiac-408 cipher and obtain slightly better (and near to optimal) results than previously published. Unlike the previous state-of-the-art approach that uses additional word lists to evaluate possible decipherments, our approach only uses a letter-based 6-gram language model. Furthermore we use our algorithm to solve large vocabulary substitution ciphers and improve the best published decipherment error rate based on the Gigaword corpus of 7.8% to 6.0% error rate.

## 1 Introduction

State-of-the-art statistical machine translation (SMT) systems use large amounts of parallel data to estimate translation models. However, parallel corpora are expensive and not available for every domain.

Recently different works have been published that train translation models using only non-parallel data. Although first practical applications of these approaches have been shown, the overall

decipherment accuracy of the proposed algorithms is still low. Improving the core decipherment algorithms is an important step for making decipherment techniques useful for practical applications.

In this paper we present an effective beam search algorithm which provides high decipherment accuracies while having low computational requirements. The proposed approach allows using high order  $n$ -gram language models, is scalable to large vocabulary sizes and can be adjusted to account for a given amount of computational resources. We show significant improvements in decipherment accuracy in a variety of experiments while being computationally more effective than previous published works.

## 2 Related Work

The experiments proposed in this paper touch many of previously published works in the decipherment field.

Regarding the decipherment of 1:1 substitution ciphers various works have been published: Most older papers do not use a statistical approach and instead define some heuristic measures for scoring candidate decipherments. Approaches like (Hart, 1994) and (Olson, 2007) use a dictionary to check if a decipherment is useful. (Clark, 1998) defines other suitability measures based on  $n$ -gram counts and presents a variety of optimization techniques like simulated annealing, genetic algorithms and tabu search.

On the other hand, statistical approaches for 1:1 substitution ciphers were published in the natural language processing community: (Ravi and Knight, 2008) solve 1:1 substitution ciphers optimally by formulating the decipherment problem as an integer linear program (ILP) while (Corlett and Penn, 2010) solve the problem using  $A^*$  search. We use our own implementation of these methods to report optimal solutions to 1:1 substitution ci-

phers for language model orders  $n = 2$  and  $n = 3$ .

(Ravi and Knight, 2011a) report the first automatic decipherment of the Zodiac-408 cipher. They use a combination of a 3-gram language model and a word dictionary. We run our beam search approach on the same cipher and report better results without using an additional word dictionary—just by using a high order  $n$ -gram language model.

(Ravi and Knight, 2011b) report experiments on large vocabulary substitution ciphers based on the Transtac corpus. (Dou and Knight, 2012) improve upon these results and provide state-of-the-art results on a large vocabulary word substitution cipher based on the Gigaword corpus. We run our method on the same corpus and report improvements over the state of the art.

(Ravi and Knight, 2011b) and (Nuhn et al., 2012) have shown that—even for larger vocabulary sizes—it is possible to learn a full translation model from non-parallel data. Even though this work is currently only able to deal with substitution ciphers, phenomena like reordering, insertions and deletions can in principle be included in our approach.

### 3 Definitions

In the following we will use the machine translation notation and denote the **ciphertext** with  $f_1^N = f_1 \dots f_j \dots f_N$  which consists of cipher tokens  $f_j \in V_f$ . We denote the **plaintext** with  $e_1^N = e_1 \dots e_i \dots e_N$  (and its vocabulary  $V_e$  respectively). We define

$$e_0 = f_0 = e_{N+1} = f_{N+1} = \$ \quad (1)$$

with “\$” being a special sentence boundary token. We use the abbreviations  $\bar{V}_e = V_e \cup \{\$\}$  and  $\bar{V}_f$  respectively.

A **general substitution cipher** uses a table  $s(e|f)$  which contains for each cipher token  $f$  a probability that the token  $f$  is substituted with the plaintext token  $e$ . Such a table for substituting cipher tokens  $\{A, B, C, D\}$  with plaintext tokens  $\{a, b, c, d\}$  could for example look like

	a	b	c	d
A	0.1	0.2	0.3	0.4
B	0.4	0.2	0.1	0.3
C	0.4	0.1	0.2	0.3
D	0.3	0.4	0.2	0.1

The **1:1 substitution cipher** encrypts a given plaintext into a ciphertext by replacing each plaintext token with a unique substitute: This means that the table  $s(e|f)$  contains all zeroes, except for one “1.0” per  $f \in V_f$  and one “1.0” per  $e \in V_e$ . For example the text

abadcab

would be enciphered to

BCBADBC

when using the substitution

	a	b	c	d
A	0	0	0	1
B	1	0	0	0
C	0	1	0	0
D	0	0	1	0

In contrast to the 1:1 substitution cipher, the **homophonic substitution cipher** allows multiple cipher tokens per plaintext token, which means that the table  $s(e|f)$  is all zero, except for one “1.0” per  $f \in V_f$ . For example the above plaintext could be enciphered to

ABCDECF

when using the homophonic substitution

	a	b	c	d
A	1	0	0	0
B	0	1	0	0
C	1	0	0	0
D	0	0	0	1
E	0	0	1	0
F	0	1	0	0

We will use the definition

$$n_{max} = \max_e \sum_f s(e|f) \quad (2)$$

to characterize the maximum number of different cipher symbols allowed per plaintext symbol.

We formalize the 1:1 substitutions with a *bijective* function  $\phi : V_f \rightarrow V_e$  and homophonic substitutions with a *general* function  $\phi : V_f \rightarrow V_e$ .

Following (Corlett and Penn, 2010), we call cipher functions  $\phi$ , for which not all  $\phi(f)$ ’s are fixed, **partial cipher functions**. Further,  $\phi'$  is **said to extend**  $\phi$ , if for all  $f$  that are fixed in  $\phi$ , it holds that  $f$  is also fixed in  $\phi'$  with  $\phi'(f) = \phi(f)$ .

The **cardinality** of  $\phi$  counts the number of fixed  $f$ 's in  $\phi$ .

When talking about partial cipher functions we use the **notation for relations**, in which  $\phi \subseteq V_f \times V_e$ . For example with

$$\phi = \{(A, a)\} \quad \phi' = \{(A, a), (B, b)\}$$

it follows that  $\phi \subseteq^1 \phi'$  and

$$\begin{array}{ll} |\phi| = 1 & |\phi'| = 2 \\ \phi(A) = a & \phi'(A) = a \\ \phi(B) = \text{undefined} & \phi'(B) = b \end{array}$$

The general **decipherment goal** is to obtain a mapping  $\phi$  such that the probability of the deciphered text is maximal:

$$\hat{\phi} = \arg \max_{\phi} p(\phi(f_1)\phi(f_2)\phi(f_3)\dots\phi(f_N)) \quad (3)$$

Here  $p(\dots)$  denotes the **language model**. Depending on the structure of the language model Equation 3 can be further simplified.

## 4 Beam Search

In this Section we present our beam search approach to solving Equation 3. We first present the general algorithm, containing many higher level functions. We then discuss possible instances of these higher level functions.

### 4.1 General Algorithm

Figure 1 shows the general structure of the beam search algorithm for the decipherment of substitution ciphers. The general idea is to keep track of all partial hypotheses in two arrays  $H_s$  and  $H_t$ . During search all possible extensions of the partial hypotheses in  $H_s$  are generated and scored. Here, the function `EXT_ORDER` chooses which cipher symbol is used next for extension, `EXT_LIMITS` decides which extensions are allowed, and `SCORE` scores the new partial hypotheses. `PRUNE` then selects a subset of these hypotheses which are stored to  $H_t$ . Afterwards the array  $H_s$  is copied to  $H_t$  and the search process continues with the updated array  $H_s$ .

Due to the structure of the algorithm the cardinality of all hypotheses in  $H_s$  increases in each step. Thus only hypotheses of the same cardinality

<sup>1</sup>shorthand notation for  $\phi'$  extends  $\phi$

---

```

1: function      BEAM_SEARCH(EXT_ORDER,
   EXT_LIMITS, PRUNE)
2:   init sets  $H_s, H_t$ 
3:   CARDINALITY = 0
4:    $H_s$ .ADD( $(\emptyset, 0)$ )
5:   while CARDINALITY <  $|V_f|$  do
6:      $f = \text{EXT\_ORDER}[\text{CARDINALITY}]$ 
7:     for all  $\phi \in H_s$  do
8:       for all  $e \in V_e$  do
9:          $\phi' := \phi \cup \{(e, f)\}$ 
10:        if EXT_LIMITS( $\phi'$ ) then
11:           $H_t$ .ADD( $\phi'$ , SCORE( $\phi'$ ))
12:        end if
13:      end for
14:    end for
15:    PRUNE( $H_t$ )
16:    CARDINALITY = CARDINALITY + 1
17:     $H_s = H_t$ 
18:     $H_t$ .CLEAR()
19:  end while
20:  return best scoring cipher function in  $H_s$ 
21: end function

```

---

Figure 1: The general structure of the beam search algorithm for decipherment of substitution ciphers. The high level functions `SCORE`, `EXT_ORDER`, `EXT_LIMITS` and `PRUNE` are described in Section 4.

are compared in the pruning step. When  $H_s$  contains full cipher relations, the cipher relation with the maximal score is returned.<sup>2</sup>

Figure 2 illustrates how the algorithm explores the search space for a homophonic substitution cipher. In the following we show several instances of `EXT_ORDER`, `EXT_LIMITS`, `SCORE`, and `PRUNE`.

### 4.2 Extension Limits (EXT\_LIMITS)

In addition to the implicit constraint of  $\phi$  being a function  $V_f \rightarrow V_e$ , one might be interested in functions of a specific form:

For 1:1 substitution ciphers (`EXT_LIMITS_SIMPLE`)  $\phi$  must fulfill that the number of cipher letters  $f \in V_f$  that map to any  $e \in V_e$  is at most one. Since partial hypotheses violating this condition can never “recover” when being extended, it becomes clear that these partial hypotheses can be left out from search.

<sup>2</sup> $n$ -best output can be implemented by returning the  $n$  best scoring hypotheses in the final array  $H_s$ .

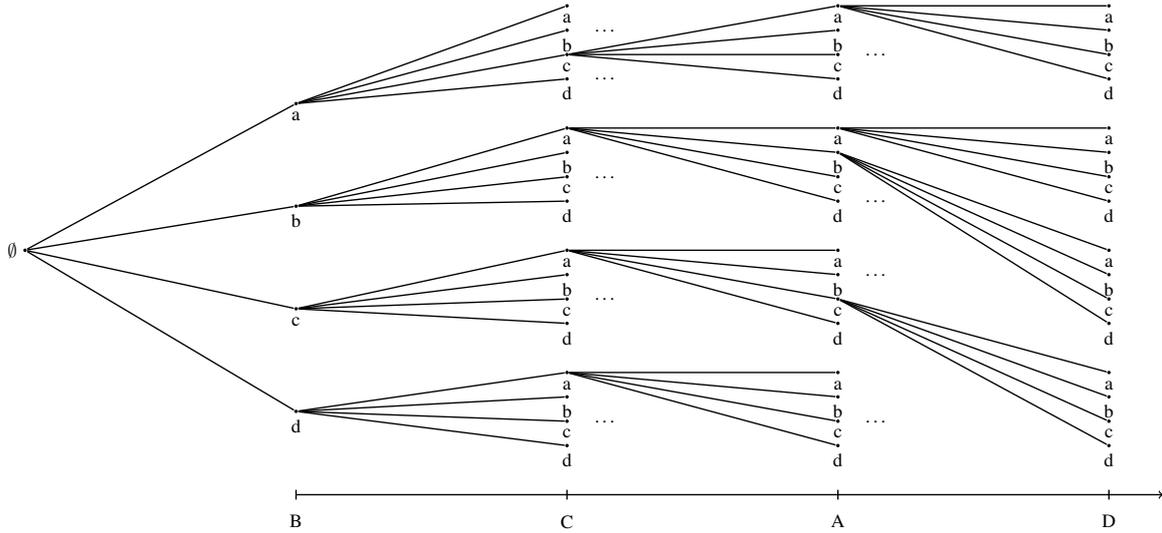


Figure 2: Illustration of the search space explored by the beam search algorithm with cipher vocabulary  $V_f = \{A, B, C, D\}$ , plaintext vocabulary  $V_e = \{a, b, c, d\}$ ,  $\text{EXT\_ORDER} = (B, C, A, D)$ , homophonic extension limits ( $\text{EXT\_LIMITS\_HOMOPHONIC}$ ) with  $n_{max} = 4$ , and histogram pruning with  $n_{keep} = 4$ . Hypotheses are visualized as nodes in the tree. The  $x$ -axis represents the extension order. At each level only those 4 hypotheses that survived the histogram pruning process are extended.

Homophonic substitution ciphers can be handled by the beam search algorithm, too. Here the condition that  $\phi$  must fulfill is that the number of cipher letters  $f \in V_f$  that map to any  $e \in V_e$  is at most  $n_{max}$  (which we will call  $\text{EXT\_LIMITS\_HOMOPHONIC}$ ). As soon as this condition is violated, all further extensions will also violate the condition. Thus, these partial hypotheses can be left out.

### 4.3 Score Estimation (SCORE)

The score estimation function needs to predict how good or bad a *partial* hypothesis (cipher function) might become. We propose simple heuristics that use the  $n$ -gram counts rather than the original ciphertext. The following formulas consider the 2-gram case. Equations for higher  $n$ -gram orders can be obtained analogously.

With Equation 3 in mind, we want to estimate the best possible score

$$\prod_{j=1}^{N+1} p(\phi'(f_j) | \phi'(f_{j-1})) \quad (4)$$

which can be obtained by extensions  $\phi' \supseteq \phi$ . By defining counts<sup>3</sup>

$$N_{ff'} = \sum_{i=1}^{N+1} \delta(f, f_{i-1}) \delta(f', f_i) \quad (5)$$

<sup>3</sup> $\delta$  denotes the Kronecker delta.

we can equivalently use the scores

$$\sum_{f, f' \in \bar{V}_f} N_{ff'} \log p(\phi'(f') | \phi'(f)) \quad (6)$$

Using this formulation it is easy to propose a whole class of heuristics: We only present the simplest heuristic, which we call  $\text{TRIVIAL\_HEURISTIC}$ . Its name stems from the fact that it only evaluates those parts of a given  $\phi'$  that are already fixed, and thus does not estimate any future costs. Its score is calculated as

$$\sum_{f, f' \in \phi'} N_{ff'} \log p(\phi'(f') | \phi'(f)). \quad (7)$$

Here  $f, f' \in \phi'$  denotes that  $f$  and  $f'$  need to be covered in  $\phi'$ . This heuristic is *optimistic* since we implicitly use “0” as estimate for the non fixed parts of the sum, for which  $N_{ff'} \log p(\cdot | \cdot) \leq 0$  holds.

It should be noted that this heuristic can be implemented very efficiently. Given a partial hypothesis  $\phi$  with given  $\text{SCORE}(\phi)$  the score of an extension  $\phi'$  can be calculated as

$$\text{SCORE}(\phi') = \text{SCORE}(\phi) + \text{NEWLY\_FIXED}(\phi, \phi') \quad (8)$$

where  $\text{NEWLY\_FIXED}$  only includes scores for  $n$ -grams that have been newly fixed in  $\phi'$  during the extension step from  $\phi$  to  $\phi'$ .

#### 4.4 Extension Order (EXT\_ORDER)

For the choice which ciphertext symbol should be fixed next during search, several possibilities exist: The overall goal is to choose an extension order that leads to an overall low error rate. Intuitively it seems a good idea to first try to decipher higher frequent words rather than the lowest frequent ones. It is also clear that the choice of a good extension order is dependent on the score estimation function SCORE: The extension order should lead to informative scores early on so that misleading hypotheses can be pruned out early.

In most of our experiments we will make use of a very simple extension order: HIGHEST\_UNIGRAM\_FREQUENCY simply fixes the most frequent symbols first.

In case of the Zodiac-408, we use another strategy that we call HIGHEST\_NGRAM\_COUNT extension order. In each step it greedily chooses the symbol that will maximize the number of fixed ciphertext  $n$ -grams. This strategy is useful because the SCORE function we use is TRIVIAL\_HEURISTIC, which is not able to provide informative scores if only few full  $n$ -grams are fixed.

#### 4.5 Pruning (PRUNE)

We propose two pruning methods: HISTOGRAM\_PRUNING sorts all hypotheses according to their score and then keeps only the best  $n_{keep}$  hypotheses.

THRESHOLD\_PRUNING keeps only those hypotheses  $\phi_{keep}$  for which

$$\text{SCORE}(\phi_{keep}) \geq \text{SCORE}(\phi_{best}) - \beta \quad (9)$$

holds for a given parameter  $\beta \geq 0$ . Even though THRESHOLD\_PRUNING has the advantage of not needing to sort all hypotheses, it has proven difficult to choose proper values for  $\beta$ . Due to this, all experiments presented in this paper only use HISTOGRAM\_PRUNING.

### 5 Iterative Beam Search

(Ravi and Knight, 2011b) propose a so called “iterative EM algorithm”. The basic idea is to run a decipherment algorithm—in their case an EM algorithm based approach—on a subset of the vocabulary. After having obtained the results from the restricted vocabulary run, these results are used to initialize a decipherment run with a larger vocabulary. The results from this run will then be used for a further decipherment run with an even

larger vocabulary and so on. In our large vocabulary word substitution cipher experiments we iteratively increase the vocabulary from the 1000 most frequent words, until we finally reach the 50000 most frequent words.

## 6 Experimental Evaluation

We conduct experiments on letter based 1:1 substitution ciphers, the homophonic substitution cipher Zodiac-408, and word based 1:1 substitution ciphers.

For a given reference mapping  $\phi_{ref}$ , we evaluate candidate mappings  $\phi$  using two error measures: **Mapping Error Rate**  $\text{MER}(\phi, \phi_{ref})$  and **Symbol Error Rate**  $\text{SER}(\phi, \phi_{ref})$ . Roughly speaking, SER reports the fraction of symbols in the deciphered text that are not correct, while MER reports the fraction of incorrect mappings in  $\phi$ .

Given a set of symbols  $V_{eval}$  with unigram counts  $N(v)$  for  $v \in V_{eval}$ , and the total amount of running symbols  $N_{eval} = \sum_{v \in V_{eval}} N(v)$  we define

$$\text{MER} = 1 - \sum_{v \in V_{eval}} \frac{1}{|V_{eval}|} \cdot \delta(\phi(v), \phi_{ref}(v)) \quad (10)$$

$$\text{SER} = 1 - \sum_{v \in V_{eval}} \frac{N(v)}{N_{eval}} \cdot \delta(\phi(v), \phi_{ref}(v)) \quad (11)$$

Thus the SER can be seen as a weighted form of the MER, emphasizing errors for frequent words. In decipherment experiments, SER will often be lower than MER, since it is often easier to decipher frequent words.

### 6.1 Letter Substitution Ciphers

As **ciphertext** we use the text of the English Wikipedia article about *History*<sup>4</sup>, remove all pictures, tables, and captions, convert all letters to lowercase, and then remove all non-letter and non-space symbols. This corpus forms the basis for shorter cryptograms of size 2, 4, 8, 16, 32, 64, 128, and 256—of which we generate 50 each. We make sure that these shorter cryptograms do not end or start in the middle of a word. We create the ciphertext using a **1:1 substitution cipher** in which we fix the mapping of the space symbol ' '. This

<sup>4</sup><http://en.wikipedia.org/wiki/History>

Order	Beam	MER [%]	SER [%]	RT [s]
3	10	33.15	25.27	0.01
	100	12.00	6.95	0.06
	1k	7.37	3.06	0.53
	10k	5.10	1.42	5.33
	100k	4.93	1.31	47.70
	$\infty^*$	4.93	1.31	19 700.00
4	10	55.97	48.19	0.02
	100	18.15	14.41	0.10
	1k	5.13	3.42	0.89
	10k	1.55	1.00	8.57
	100k	0.39	0.06	81.34
5	10	69.19	60.13	0.02
	100	35.57	29.02	0.14
	1k	10.89	8.47	1.29
	10k	0.38	0.06	11.91
	100k	0.38	0.06	120.38
6	10	74.65	64.77	0.03
	100	40.26	33.38	0.17
	1k	13.53	10.08	1.58
	10k	2.45	1.28	15.77
	100k	0.09	0.05	151.85

Table 1: Symbol error rates (SER), Mapping error rates (MER) and runtimes (RT) in dependence of language model order (ORDER) and histogram pruning size (BEAM) for decipherment of letter substitution ciphers of length 128. Runtimes are reported on a single core machine. Results for beam size “ $\infty$ ” were obtained using  $A^*$  search.

makes our experiments comparable to those conducted in (Ravi and Knight, 2008). Note that fixing the ‘\_’ symbol makes the problem much easier: The exact methods show much higher computational demands for lengths beyond 256 letters when not fixing the space symbol.

The **plaintext language model** we use a letter based ( $V_e = \{a, \dots, z, _\}$ ) language model trained on a subset of the Gigaword corpus (Graff et al., 2007).

We use **extension limits** fitting the 1:1 substitution cipher  $n_{max} = 1$  and **histogram pruning** with different beam sizes.

For comparison we reimplemented the ILP approach from (Ravi and Knight, 2008) as well as the  $A^*$  approach from (Corlett and Penn, 2010).

Figure 3 shows the **results** of our algorithm for different cipher length. We use a beam size of 100k for the 4, 5 and 6-gram case. Most remarkably our 6-gram beam search results are significantly better than all methods presented in the literature. For the cipher length of 32 we obtain a symbol error rate of just 4.1% where the optimal solution (i.e. without search errors) for a 3-gram

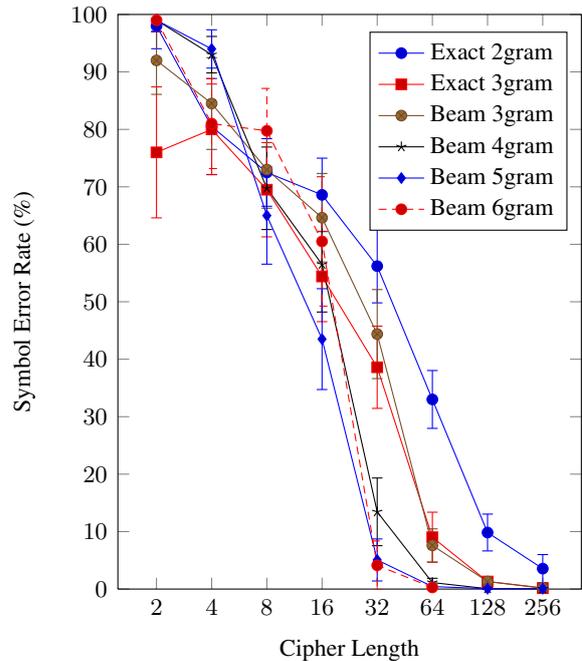


Figure 3: Symbol error rates for decipherment of letter substitution ciphers of different lengths. Error bars show the 95% confidence interval based on decipherment on 50 different ciphers. Beam search was performed with a beam size of “100k”.

language model has a symbol error rate as high as 38.3%.

Table 1 shows error rates and runtimes of our algorithm for different beam sizes and language model orders given a fixed ciphertext length of 128 letters. It can be seen that achieving close to optimal results is possible in a fraction of the CPU time needed for the optimal solution: In the 3-gram case the optimal solution is found in  $\frac{1}{400}$ th of the time needed using  $A^*$  search. It can also be seen that increasing the language model order does not increase the runtime much while providing better results if the beam size is large enough: If the beam size is not large enough, the decipherment accuracy decreases when increasing the language model order: This is because the higher order heuristics do not give reliable scores if only few  $n$ -grams are fixed.

To summarize: The beam search method is significantly faster and obtains significantly better results than previously published methods. Furthermore it offers a good trade-off between CPU time and decipherment accuracy.

i l i k e k i l l i n g p e o p l  
 Δ ▣ P / Z / U B ▣ X O R X 9 X X B  
 e b e c a u s e i t i s s o m u c  
 w V + 3 G Y F O Δ H P ▣ K 3 P Y 3  
 h f u n i t i n m o r e f u n t h  
 M J Y Λ U I X Δ P T L N G Y D ● ⊖  
 a n k i l l i n g w i l d g a m e  
 S † / Δ ▣ B P O R A U ▣ 7 R J P E  
 i n t h e f o r r e s t b e c a u  
 X Λ L M Z J Q R \ 9 F H V w 3 Δ Y  
 s e m a n i s t h e m o a t r a n  
 ▣ + P G D Δ K I ⊖ O P X Δ ● † S †  
 g e r o u e a n a m a l o f a l l  
 R N L I Y E J O Δ P G B T G S ▣ B  
 t o k i l l s o m e t h i n g g i  
 L Q / P ▣ B ▣ X P E H M U Λ R R X

Figure 4: First 136 letters of the Zodiac-408 cipher and its decipherment.

## 6.2 Zodiac-408 Cipher

As **ciphertext** we use a transcription of the Zodiac-408 cipher. It consists of 54 different symbols and has a length of 408 symbols.<sup>5</sup> The cipher has been deciphered by hand before. It contains some mistakes and ambiguities: For example, it contains misspelled words like *forrest* (vs. *forest*), *experence* (vs. *experience*), or *paradice* (vs. *paradise*). Furthermore, the last 17 letters of the cipher do not form understandable English when applying the same homophonic substitution that decipheres the rest of the cipher. This makes the Zodiac-408 a good candidate for testing the robustness of a decipherment algorithm.

We assume a **homophonic substitution cipher**, even though the cipher is not strictly homophonic: It contains three cipher symbols that correspond to two or more plaintext symbols. We ignore this fact for our experiments, and count—in case of the MER *only*—the decipherment for these symbols as correct when the obtained mapping is contained in the set of reference symbols. We use **extension limits** with  $n_{max} = 8$  and **histogram pruning** with beam sizes of  $10k$  up to  $10M$ .

The **plaintext language model** is based on the same subset of Gigaword (Graff et al., 2007) data as the experiments for the letter substitution ciphers. However, we first removed all space sym-

<sup>5</sup>hence its name

Order	Beam	MER [%]	SER [%]	RT [s]
4	10k	71.43	67.16	222
	100k	66.07	61.52	1 460
	1M	39.29	34.80	12 701
	10M	19.64	16.18	125 056
5	10k	94.64	96.57	257
	100k	10.71	5.39	1 706
	1M	8.93	3.19	14 724
	10M	8.93	3.19	152 764
6	10k	87.50	84.80	262
	100k	94.64	94.61	1 992
	1M	8.93	2.70	17 701
	<b>10M</b>	<b>7.14</b>	<b>1.96</b>	167 181

Table 2: Symbol error rates (SER), Mapping error rates (MER) and runtimes (RT) in dependence of language model order (ORDER) and histogram pruning size (BEAM) for the decipherment of the Zodiac-408 cipher. Runtimes are reported on a 128-core machine.

bols from the training corpus before training the actual letter based 4-gram, 5-gram, and 6-gram language model on it. Other than (Ravi and Knight, 2011a) we do not use any word lists and by that avoid any degrees of freedom in how to integrate it into the search process: Only an  $n$ -gram language model is used.

Figure 4 shows the first parts of the cipher and our best decipherment. Table 2 shows the **results** of our algorithm on the Zodiac-408 cipher for different language model orders and pruning settings.

To summarize: Our final decipherment—for which we only use a 6-gram language model—has a symbol error rate of only 2.0%, which is slightly better than the best decipherment reported in (Ravi and Knight, 2011a). They used an  $n$ -gram language model together with a word dictionary and obtained a symbol error rate of 2.2%. We thus obtain better results with less modeling.

## 6.3 Word Substitution Ciphers

As **ciphertext**, we use parts of the JRC corpus (Steinberger et al., 2006) and the Gigaword corpus (Graff et al., 2007). While the full JRC corpus contains roughly 180k word types and consists of approximately 70M running words, the full Gigaword corpus contains around 2M word types and roughly 1.5G running words.

We run experiments for three different setups: The “JRC” and “Gigaword” setups use the first half of the respective corpus as ciphertext, while the **plaintext language model** of order  $n = 3$  was

Setup	Top	MER [%]	SER [%]	RT [hh:mm]
Gigaword	1k	81.91	27.38	03h 10m
	10k	30.29	8.55	09h 21m
	20k	21.78	6.51	16h 25m
	50k	19.40	5.96	49h 02m
JRC	1k	73.28	15.42	00h 32m
	10k	15.82	2.61	13h 03m
JRC-Shuf	1k	76.83	19.04	00h 31m
	10k	15.08	2.58	13h 03m

Table 3: Word error rates (WER), Mapping error rates (MER) and runtimes (RT) for iterative decipherment run on the (TOP) most frequent words. Error rates are evaluated on the full vocabulary. Runtimes are reported on a 128-core machine.

trained on the second half. The “JRC-Shuf” setup is created by randomly selecting half of the sentences of the JRC corpus as ciphertext, while the language model was trained on the complementary half of the corpus.

We encrypt the ciphertext using a **1:1 substitution cipher** on word level, imposing a much larger vocabulary size. We use **histogram pruning** with a beam size of 128 and use **extension limits** of  $n_{max} = 1$ . Different to the previous experiments, we use **iterative beam search** with iterations as shown in Table 3.

The results for the Gigaword task are directly comparable to the word substitution experiments presented in (Dou and Knight, 2012). Their final decipherment has a symbol error rate of 7.8%. Our algorithm obtains 6.0% symbol error rate. It should be noted that the improvements of 1.8% symbol error rate correspond to a larger improvement in terms of mapping error rate. This can also be seen when looking at Table 3: An improvement of the symbol error rate from 6.51% to 5.96% corresponds to an improvement of mapping error rate from 21.78% to 19.40%.

To summarize: Using our beam search algorithm in an iterative fashion, we are able to improve the state-of-the-art decipherment accuracy for word substitution ciphers.

## 7 Conclusion

We have presented a simple and effective beam search approach to the decipherment problem. We have shown in a variety of experiments—letter substitution ciphers, the Zodiac-408, and word substitution ciphers—that our approach outperforms the current state of the art while being con-

ceptually simpler and keeping computational demands low.

We want to note that the presented algorithm is not restricted to 1:1 and homophonic substitution ciphers: It is possible to extend the algorithm to solve  $n:m$  mappings. Along with more sophisticated pruning strategies, score estimation functions, and extension orders, this will be left for future research.

## Acknowledgements

This work was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation. Experiments were performed with computing resources granted by JARA-HPC from RWTH Aachen University under project “jara0040”.

## References

- Andrew J. Clark. 1998. *Optimisation heuristics for cryptology*. Ph.D. thesis, Faculty of Information Technology, Queensland University of Technology.
- Eric Corlett and Gerald Penn. 2010. An exact A\* method for deciphering letter-substitution ciphers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1040–1047, Uppsala, Sweden, July. The Association for Computer Linguistics.
- Qing Dou and Kevin Knight. 2012. Large scale decipherment for out-of-domain machine translation. In *Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 266–275, Jeju Island, Korea, July. Association for Computational Linguistics.
- David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English Gigaword Third Edition. Linguistic Data Consortium, Philadelphia.
- George W. Hart. 1994. To decode short cryptograms. *Communications of the Association for Computing Machinery (CACM)*, 37(9):102–108, September.
- Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 156–164, Jeju, Republic of Korea, July. Association for Computational Linguistics.
- Edwin Olson. 2007. Robust dictionary attack of short simple substitution ciphers. *Cryptologia*, 31(4):332–342, October.

- Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 812–819, Honolulu, Hawaii. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2011a. Bayesian inference for Zodiac and other homophonic ciphers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 239–247, Portland, Oregon, June. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2011b. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 12–21, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, and Dan Tufiş. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, pages 2142–2147. European Language Resources Association.

# Social Text Normalization using Contextual Graph Random Walks

**Hany Hassan**

Microsoft Research  
Redmond, WA

hanyh@microsoft.com

**Arul Menezes**

Microsoft Research  
Redmond, WA

arulm@microsoft.com

## Abstract

We introduce a social media text normalization system that can be deployed as a preprocessing step for Machine Translation and various NLP applications to handle social media text. The proposed system is based on unsupervised learning of the normalization equivalences from unlabeled text. The proposed approach uses Random Walks on a contextual similarity bipartite graph constructed from n-gram sequences on large unlabeled text corpus. We show that the proposed approach has a very high precision of (92.43) and a reasonable recall of (56.4). When used as a preprocessing step for a state-of-the-art machine translation system, the translation quality on social media text improved by 6%. The proposed approach is domain and language independent and can be deployed as a preprocessing step for any NLP application to handle social media text.

## 1 Introduction

Social Media text is usually very noisy and contains a lot of typos, ad-hoc abbreviations, phonetic substitutions, customized abbreviations and slang language. The social media text is evolving with new entities, words and expressions. Natural language processing and understanding systems such as Machine Translation, Information Extraction and Text-to-Speech are usually trained and optimized for clean data; therefore such systems would face a challenging problem with social media text.

Various social media genres developed distinct characteristics. For example, SMS developed a nature of shortening messages to avoid multiple keystrokes. On the other hand, Facebook and instant messaging developed another genre where

more emotional expressions and different abbreviations are very common. Somewhere in between, Twitter's statuses come with some brevity similar to SMS along with the social aspect of Facebook. On the same time, various social media genres share many characteristics and typologies. For example, repeating letters or punctuation for emphasizing and emotional expression such as "*gooooood morniiiiing*". Using phonetic spelling in a generalized way or to reflect a local accent; such as "*wuz up bro*" (*what is up brother*). Eliminating vowels such as "*cm to c my luv*". Substituting numbers for letters such as "*4get*" (*forget*), "*2morrow*" (*tomorrow*), and "*b4*" (*before*). Substituting phonetically similar letters such as "*phone*" (*fon*). Slang abbreviations which usually abbreviates multi-word expression such as "*LMS*" (*like my status*), "*idk*" (*i do not know*), "*rofl*" (*rolling on floor laughing*).

While social media genres share many characteristics, they have significant differences as well. It is crucial to have a solution for text normalization that can adapt to such variations automatically. We propose a text normalization approach using an unsupervised method to induce normalization equivalences from noisy data which can adapt to any genre of social media.

In this paper, we focus on providing a solution for social media text normalization as a preprocessing step for NLP applications. However, this is a challenging problem for several reasons. First, it is not straightforward to define the Out-of-Vocabulary (OOV) words. Traditionally, an OOV word is defined as a word that does not exist in the vocabulary of a given system. However, this definition is not adequate for the social media text which has a very dynamic nature. Many words and named entities that do not exist in a given vocabulary should not be considered for normalization. Second, same OOV word may have many

appropriate normalization depending on the context and on the domain. Third, text normalization as a preprocessing step should have very high precision; in other words, it should provide conservative and confident normalization and not overcorrect. Moreover, the text normalization should have high recall, as well, to have a good impact on the NLP applications.

In this paper, we introduce a social media text normalization system which addresses the challenges mentioned above. The proposed system is based on constructing a lattice from possible normalization candidates and finding the best normalization sequence according to an n-gram language model using a Viterbi decoder. We propose an unsupervised approach to learn the normalization candidates from unlabeled text data. The proposed approach uses Random Walks on a contextual similarity graph constructed from n-gram sequences on large unlabeled text corpus. The proposed approach is very scalable, accurate and adaptive to any domain and language. We evaluate the approach on the normalization task as well as machine translation task.

The rest of this paper is organized as follows: Section(2) discusses the related work, Section(3) introduces the text normalization system and the baseline candidate generators, Section(4) introduces the proposed graph-based lexicon induction approach, Section(5) discusses the experiments and output analysis, and finally Section(6) concludes and discusses future work.

## 2 Related Work

Early work handled the text normalization problem as a noisy channel model where the normalized words go through a noisy channel to produce the noisy text. (Brill and Moore, 2000) introduced an approach for modeling the spelling errors as a noisy channel model based on string to string edits. Using this model gives significant performance improvements compared to previously proposed models. (Toutanova and Moore, 2002) improved the string to string edits model by modeling pronunciation similarities between words. (Choudhury et al., 2007) introduced a supervised HMM channel model for text normalization which has been expanded by (Cook and Stevenson, 2009) to introduce unsupervised noisy channel model using probabilistic models for common abbreviation and various spelling errors types. Some

researchers used Statistical Machine Translation approach for text normalization; formalizing the problem as a translation from the noisy forms to the normalized forms. (Aw et al., 2006) proposed an approach for normalizing Short Messaging Service (SMS) texts by translating it into normalized forms using Phrase-based SMT techniques on character level. The main drawback of these approaches is that the noisy channel model cannot accurately represent the errors types without contextual information.

More recent approaches tried to handle the text normalization problem using normalization lexicons which map the noisy form of the word to a normalized form. For example, (Han et al., 2011) proposed an approach using a classifier to identify the noisy words candidate for normalization; then using some rules to generate lexical variants and a small normalization lexicon. (Gouws et al., 2011) proposed an approach using an impoverished normalization lexicon based on string and distributional similarity along with a dictionary lookup approach to detect noisy words. More recently, (Han et al., 2012) introduced a similar approach by generating a normalization lexicon based on distributional similarity and string similarity. This approach uses pairwise similarity where any two words that share the same context are considered as normalization equivalences. The pairwise approach has a number of limitations. First, it does not take into account the relative frequencies of the normalization equivalences that might share different contexts. Therefore, the selection of the normalization equivalences is performed on pairwise basis only and is not optimized over the whole data. Secondly, the normalization equivalences must appear in the exact same context to be considered as a normalization candidate. These limitations affect the accuracy and the coverage of the produced lexicon.

Our approach also adopts a lexicon based approach for text normalization, we construct a lattice from possible normalization candidates and find the best normalization sequence according to an n-gram language model using a Viterbi decoder. The normalization lexicon is acquired from unlabeled data using random walks on a contextual similarity graph constructed from n-gram sequences on large unlabeled text corpus. Our approach has some similarities with (Han et al., 2012) since both approaches utilize a normaliza-

tion lexicon acquired from unlabeled data using distributional and string similarities. However, our approach is significantly different since we acquire the lexicon using random walks on a contextual similarity graph which has a number of advantages over the pairwise similarity approach used in (Han et al., 2012). Namely, the acquired normalization equivalences are optimized globally over the whole data, the rare equivalences are not considered as good candidates unless there is a strong statistical evidence across the data, and finally the normalization equivalences may not share the same context. Those are clear advantages over the pairwise similarity approach and result in a lexicon with higher accuracy as well as wider coverage. Those advantages will be clearer when we describe the proposed approach in details and during evaluation and comparison to the pairwise approach.

### 3 Text Normalization System

In this paper, we handle text normalization as a lattice scoring approach, where the translation is performed from noisy text as the source side to the normalized text as the target side. Unlike conventional MT systems, the translation table is not learned from parallel aligned data; instead it is modeled by the graph-based approach of lexicon generation as we will describe later. We construct a lattice from possible normalization candidates and find the best normalization sequence according to an n-gram language model using a Viterbi decoder.

In this paper, we restrict the normalization lexicon to one-to-one word mappings, we do not consider multi words mapping for the lexicon induction. To identify OOV candidates for normalization; we restrict proposing normalization candidates to the words that we have in our induced normalization lexicon only. This way, the system would provide more confident and conservative normalization. We move the problem of identifying OOV words to training time; at training time we use soft criteria to identify OOV words.

#### 3.1 Baseline Normalization Candidates Generation

We experimented with two normalization candidate generators as baseline systems. The first is a dictionary based spelling correction similar to Aspell<sup>1</sup>. In this experiment we used the spell checker

<sup>1</sup><http://aspell.net/>

to generate all possible candidates for OOV words and then applied the Viterbi decoder on the constructed lattice to score the best correction candidates using a language model.

Our second candidates generator is based on a trie approximate string matching with  $K$  errors similar to the approach proposed in (Chang et al., 2010), where  $K$  errors can be caused by substitution, insertion, or deletion operations. In our implementation, we customized the errors operations to accommodate the nature of the social media text. Such as lengthening, letter substitution, letter-number substitution and phonetic substitution. This approach overcomes the main problem of the dictionary-based approach which is providing inappropriate normalization candidates to the errors styles in the social media text.

As we will show in the experiments in Section(5), dictionary-based normalization methods proved to be inadequate for social media domain normalization for many reasons. First, they provide generic corrections which are inappropriate for social media text. Second, they usually provide corrections with the minimal edit distance for any word or named entity regardless of the nature of the words. Finally, the previous approaches do not take into account the dynamics of the social media text where new terms can be introduced on a daily basis.

### 4 Normalization Lexicons using Graph-based Random Walks

#### 4.1 Bipartite Graph Representation

The main motivation of this approach is that normalization equivalences share similar context; which we call contextual similarity. For instance, assume 5-gram sequences of words, two words may be normalization equivalences if their n-gram context shares the same two words on the left and the same two words on the right. In other words, they are sharing a wild card pattern such as ( $word_1 word_2 * word_4 word_5$ ).

This contextual similarity can be represented as a bipartite graph with the first partite representing the words and the second partite representing the n-gram contexts that may be shared by words. A word node can be either normalized word or noisy word. Identifying if a word is normalized or noisy (candidate for normalization) is crucial since this decision limits the candidate noisy words to be normalized. We adopted a soft criteria for iden-

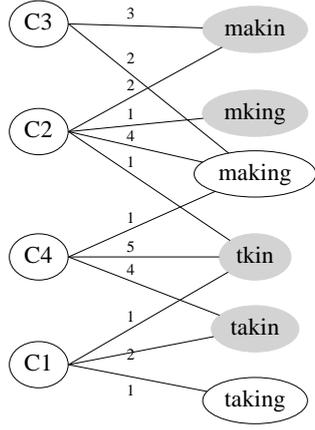


Figure 1: Bipartite Graph Representation, left nodes represent contexts, gray right nodes represent the noisy words and white right nodes represent the normalized words. Edge weight is the co-occurrence count of a word and its context.

tifying noisy words. A vocabulary is constructed from a large clean corpus. Any word that does not appear in this vocabulary more than a predefined threshold (i.e. 10 times) is considered as a candidate for normalization (noisy word). Figure(1) shows a sample of the bipartite graph  $G(W, C, E)$ , where noisy words are shown as gray nodes.

---

**Algorithm 4.1:** CONSTRUCTBIPARTITE(*text*)

---

**comment:** Construct Bipartite Graph  
**output** ( $G(W, C, E)$ )  
**comment:** Extract all  $n$ -gram sequences  
 $Ngrams \leftarrow \text{EXTRACTNGRAMS}(TextCorpus)$   
**for each**  $n \in Ngrams$   
  **do**  
    **comment:** Check for center word  
    **if** ISNOISY(*CenterWord*)  
       $W \leftarrow \text{ADDSOURCENODE}(\text{CenterWord})$   
    **else**  
       $W \leftarrow \text{ADDABSORBINGNODE}(\text{CenterWord})$   
    **comment:** add the context pattern  
     $C \leftarrow \text{ADD}(\text{Context})$   
    **comment:** edge weight  
     $E \leftarrow \text{ADD}(\text{Context}, \text{Word}, \text{count})$

---

The bipartite graph,  $G(W, C, E)$ , is composed of  $W$  which includes all nodes representing normalized words and noisy words,  $C$  which includes all nodes representing shared context, and finally  $E$  which represents the edges of the graph connecting word nodes and context nodes. The weight on the edge is simply the number of occurrences of a given word in a context. While constructing the graph, we identify if a node represents a

noisy word ( $N$ ) (called source node) or a normalized word ( $M$ ) (called absorbing node). The bipartite graph is constructed using the procedure in Algorithm(4.1).

## 4.2 Lexicon generation using Random Walks

Our proposed approach uses Markov Random Walks on the bipartite graph in Figure(1) as defined in (Norris, 1997). The main objective is to identify pairs of noisy and normalized words that can be considered as normalization equivalences. In principal, this is similar to using random walks for semi-supervised label propagation which has been introduced in (Szummer and Jaakkola, 2002) and then used in many other applications. For example, (Hughes and Ramage, 2007) used random walks on Wordnet graph to measure lexical semantic relatedness between words. (Das and Petrov, 2011) used graph-based label propagation for cross-lingual knowledge transfers to induce POS tags between two languages. (Minkov and Cohen, 2012) introduced a path constrained graph walk algorithm given a small number of labeled examples to assess nodes relatedness in the graph. In this paper, we apply the label propagation approach to the text normalization problem.

Consider a random walk on the bipartite graph  $G(W, C, E)$  starting at a noisy word (source node) and ending at a normalized word (absorbing node). The walker starts from any source node  $N_i$  belonging to the noisy words then move to any other connected node  $M_j$  with probability  $P_{ij}$ . The transition between each pair of nodes is defined by a transition probability  $P_{ij}$  which represents the normalized probability of the co-occurrence counts of the word and the corresponding context. Though the counts are symmetric, the probability is not symmetric. This is due to the probability normalization which is done according to the nodes connectivity. Therefore, the transition probability between any two nodes  $i, j$  is defined as:

$$P_{ij} = W_{ij} / \sum_{\forall k} W_{ik} \quad (1)$$

For any non-connected pair of nodes,  $P_{ij} = 0$ . It is worth noting that due to the bipartite graph representation; any word node, either noisy (source) or normalized (absorbing), is only connected to context nodes and not directly connected to any other word node.

The algorithm repeats independent random walks for  $K$  times where the walks traverse the graph randomly according to the transition probability distribution in Eqn(1); each walk starts from the source noisy node and ends at an absorbing normalized node, or consumes the maximum number of steps without hitting an absorbing node.

For any random walk the number of steps taken to traverse between any two nodes is called the hitting time (Norris, 1997). Therefore, the hitting time between a noisy and a normalized pair of nodes  $(n, m)$  with a walk  $r$  is  $h_r(n, m)$ . We define the cost between the two nodes as the average hitting time  $H(n, m)$  of all walks that connect those two nodes:

$$H(n, m) = \sum_{\forall r} h_r(n, m) / R \quad (2)$$

Consider the bipartite graph in Figure(1), assume a random walk starting at the source node representing the noisy word "tkin" then moves to the context node  $C1$  then to the absorbing node representing the normalized word "taking". This random walk will associate "tkin" with "taking" with a walk of two steps (hits). Another random walk that can connect the two words is ["tkin"  $\rightarrow$   $C4$   $\rightarrow$  "takin"  $\rightarrow$   $C1$   $\rightarrow$  "taking"], which has 4 steps (hits). In this case, the cost of this pair of nodes is the average number of hits connecting them which is 3.

It is worth noting that the random walks are selected according to the transition probability in Eqn(1); therefore, the more probable paths will be picked more frequently. The same pair of nodes can be connected with many walks of various steps (hits), and the same noisy word can be connected to many other normalized words.

We define the contextual similarity probability of a normalization equivalence pair  $n, m$  as  $L(n, m)$ . Which is the relative frequency of the average hitting of those two nodes,  $H(n, m)$ , and all other normalized nodes linked to that noisy word. Thus  $L(n, m)$ , is calculated as:

$$L(n, m) = H(n, m) / \sum_i H(n, m_i) \quad (3)$$

Furthermore, we add another similarity cost between a noisy word and a normalized word based on the lexical similarity cost,  $SimCost(n, m)$ , which we will describe in the next section. The final cost associated with a pair is:

$$Cost(n, m) = \lambda_1 L(n, m) + \lambda_2 SimCost(n, m) \quad (4)$$

---

**Algorithm 4.2:** INDUCELEXICON( $G$ )

---

```

output ( $Lexicon$ )
INIT( $Lexicon$ )
for each  $n \in W \in G(W, C, E)$ 
do
  comment: for noisy nodes only
  if ISNOISY( $n$ )
    INIT( $Rn$ )
    comment: do K random walks
    for  $i \leftarrow 0$  to  $K$ 
      do
         $Rn \leftarrow$  RANDOMWALK( $n$ )
        comment: Calculate Avg. hits and normalize
         $Ln \leftarrow$  NORMALIZE( $Rn$ )
    comment: Calculate Lexical Sim Cost
     $Ln \leftarrow$  SIMCOST( $Ln$ )
     $Ln \leftarrow$  PRUNE( $Ln$ )
   $Lexicon \leftarrow$  ADD( $Ln$ )

```

---

We used uniform interpolation, both  $\lambda_1$  and  $\lambda_2$  equals 1. The final Lexicon is constructed using those entries and if needed we prune the list to take top N according to the cost above. The algorithm is outlined in 4.2.

### 4.3 Lexical Similarity Cost

We use a similarity function proposed in (Contractor et al., 2010) which is based on Longest Common Subsequence Ratio (LCSR) (Melamed, 1999). This cost function is defined as the ratio of LCSR and Edit distance between two strings as follows:

$$SimCost(n, m) = LCSR(n, m) / ED(n, m) \quad (5)$$

$$LCSR(n, m) = LCS(n, m) / MaxLenght(n, m) \quad (6)$$

We have modified the Edit Distance calculation  $ED(n, m)$  to be more adequate for social media text. The edit distance is calculated between the consonant skeleton of the two words; by removing all vowels, we used Editex edit distance as proposed in (Zobel and Philip, 1996), repetition is reduced to a single letter before calculating the edit distance, and numbers in the middle of words are substituted by their equivalent letters.

## 5 Experiments

### 5.1 Training and Evaluation Data

We collected large amount of social media data to generate the normalization lexicon using the ran-

dom walk approach. The data consists of 73 million Twitter statuses. All tweets were collected from March/April 2012 using the Twitter Streaming APIs<sup>2</sup>. We augmented this data with 50 million sentences of clean data from English LDC Gigaword corpus<sup>3</sup>. We combined both data, noisy and clean, together to induce the normalization dictionary from them. While the Gigaword clean data was used to train the language model to score the normalized lattice.

We constructed a test set of 1000 sentences of social media which had been corrected by a native human annotator, the main guidelines were to normalize noisy words to its corresponding clean words in a consistent way according to the evidences in the context. We will refer to this test set as SM-Test. Furthermore, we developed a test set for evaluating the effect of the normalization system when used as a preprocessing step for Machine translation. The machine translation test set is composed of 500 sentences of social media English text translated to normalized Spanish text by a bi-lingual translator.

## 5.2 Evaluating Normalization Lexicon Generation

We extracted 5-gram sequences from the combined noisy and clean data; then we limited the space of noisy 5-gram sequences to those which contain only one noisy word as the center word and all other words, representing the context, are not noisy. As we mentioned before, we identify whether the word is noisy or not by looking up a vocabulary list constructed from clean data. In these experiments, the vocabulary is constructed from the Language Model data (50M sentences of the English Gigaword corpus). Any word that appears less than 10 times in this vocabulary is considered noisy and candidate for normalization during the lexicon induction process. It is worth noting that our notion of noisy word does not mean it is an OOV that has to be corrected; instead it indicates that it is candidate for correction but may be opted not to be normalized if there is no confident normalization for it. This helps to maintain the approach as a high precision text normalization system which is highly preferable as an NLP preprocessing step.

We constructed a lattice using normalization

candidates and score the best Viterbi path with 5-gram language model. We experimented with two candidate generators as baseline systems, namely the dictionary-based spelling correction and the trie approximate match with  $K$  errors; where  $K=3$ . For both candidate generators the cost function for a given candidate is calculated using the lexical similarity cost in Eqn(5). We compared those approaches with our newly proposed unsupervised normalization lexicon induction; for this case the cost for a candidate is the combined cost of the contextual similarity probability and the lexical similarity cost as defined in Eqn(4). We examine the effect of data size and the steps of the random walks on the accuracy and the coverage of the induced dictionary.

We constructed the bipartite graph with the n-gram sequences as described in Algorithm 4.1. Then the Random Walks Algorithm in 4.2 is applied with 100 walks. The total number of word nodes is about 7M nodes and the total number of context nodes is about 480M nodes. We used MapReduce framework to implement the proposed technique to handle such large graph. We experimented with the maximum number of random walk steps of 2, 4 and 6; and with different portions of the data as well. Finally, we pruned the lexicon to keep the top 5 candidates per noisy word.

Table(1) shows the resulting lexicons from different experiments.

Lexicon	Lexicon	Data	Steps
Lex1	123K	20M	4
Lex2	281K	73 M	2
Lex3	327K	73M	4
Lex4	363K	73M	6

Table 1: Generated Lexicons, steps are the Random Walks maximum steps.

As shown in Table(1), we experimented with different data sizes and steps of the random walks. The more data we have the larger the lexicon we get. Also larger steps increase the induced lexicon size. A random walk step size of 2 means that the noisy/normalized pair shares the same context; while a step size of 4 or more means that they may not share the same context. Next, we will examine the effect of lexicon size on the normalization task.

<sup>2</sup><https://dev.twitter.com/docs/streaming-apis>

<sup>3</sup><http://www ldc.upenn.edu/Catalog/LDC2011T07>

### 5.3 Text Normalization Evaluation

We experimented different candidate generators and compared it to the unsupervised lexicon approach. Table(2) shows the precision and recall on a the SM-Test set.

System	Candidates	Precision	Recall	F-Measure
Base1	Dict	33.9	15.1	20.98
Base2	Trie	26.64	27.65	27.13
RW1	Lex1	88.76	59.23	71.06
RW2	Lex2	90.66	54.06	67.73
<b>RW3</b>	<b>Lex3</b>	<b>92.43</b>	<b>56.4</b>	<b>70.05</b>
RW4	Lex4	90.87	60.73	72.8

Table 2: Text Normalization with different lexicons

In Table(2), the first baseline is using a dictionary based spell checker; which gets low precision and very low recall. Similarly the trie approximate string match is doing a similar job with better recall though the precision is worst. Both of the baseline approaches are inadequate for social media text since both will try to correct any word that is similar to a word in the dictionary. The Trie approximate match is doing better job on the recall since the approximate match is based on phonetic and lexical similarities.

On the other hand, the induced normalization lexicon approach is doing much better even with a small amount of data as we can see with system RW1 which uses Lex1 generated from 20M sentences and has 123K lexicon entry. Increasing the amount of training data does impact the performance positively especially the recall. On the other hand, increasing the number of steps has a good impact on the recall as well; but with a considerable impact on the precision. It is clear that increasing the amount of data and keeping the steps limit at "4" gives better precision and coverage as well. This is a preferred setting since the main objective of this approach is to have better precision to serve as a reliable preprocessing step for Machine Translation and other NLP applications.

### 5.4 Comparison with Pairwise Similarity

We present experimental results to compare our proposed approach with (Han et al., 2012) which used pairwise contextual similarity to induce a normalization lexicon of 40K entries, we will refer to this lexicon as HB-Dict. We compare the performance of HB-Dict and our induced dictionary (system RW3). We evaluate both system on SM-

Test test set and on (Han et al., 2012) test set of 548 sentences which we call here HB-Test.

System	Precision	Recall	F-Measure
SM-Test			
HB-Dict	71.90	26.30	38.51
RW3	92.43	56.4	70.05
HB-Test			
HB-Dict	70.0	17.9	26.3
RW3	85.37	56.4	69.93

Table 3: Text Normalization Results

As shown in Table(3), RW3 system significantly outperforms HB-Dict system with the lexicon from (Han et al., 2012) on both test sets for both precision and recall. The contextual graph random walks approach helps in providing high precision lexicon since the sampling nature of the approach helps in filtering out unreliable normalization equivalences. The random walks will traverse more frequent paths; which would lead to more probable normalization equivalence. On the other hand, the proposed approach provides high recall as well which is hard to achieve with higher precision. Since the proposed approach deploys random walks to sample paths that can traverse many steps, this relaxes the constraints that the normalization equivalences have to share the same context. Instead a noisy word may share a context with another noisy word which in turn shares a context with a clean equivalent normalization word. Therefore, we end up with a lexicon that have much higher recall than the pairwise similarity approach since it explores equivalences beyond the pairwise relation. Moreover, the random walk sampling emphasis the more frequent paths and hence provides high precision lexicon.

### 5.5 Output Analysis

Table(4) shows some examples of the induced normalization equivalences, the first part shows good examples where vowels are restored and phonetic similar words are matched. Remarkably the correction "viewability" to "visibility" is interesting since the system picked the more frequent form. Moreover, the lexicon contains some entries with foreign language words normalized to its English translation. On the other hand, the lexicon has some bad normalization such as "unrecycled" which should be normalized to "non recycled" but since the system is limited to one word correction it did not get it. Another interesting bad normalization is "tutting" which is new type of

dancing and should not be corrected to "tweeting".

Noisy	Clean	Remarks
tnght	tonight	Vowels restored
darlin	darling	g restored
urung	orange	phonetic similarity
viewability	visibility	good correction
unrecycled	recycled	negation ignored
tutting	tweeting	tutting is dancing type

Table 4: Lexicon Samples

Table 5 lists a number of examples and their normalization using both **Baseline1** and **RW3**. At the first example, **RW3** got the correct normalization as "interesting" which apparently is not the one with the shortest edit distance, though it is the most frequent candidate at the generated lexicon. The baseline system did not get it right; it got a wrong normalization with shorter edit distance. Example(2) shows the same effect by getting "cuz" normalized to "because". At Example(3), both the baseline and RW3 did not get the correct normalization of "yur" to "you are" which is currently a limitation in our system since we only allow one-to-one word mapping in the generated lexicons not one-to-many or many-to-many. At Example(4), RW3 did not normalize "dure" to "sure"; however the baseline normalized it by mistake to "dare". This shows a characteristic of the proposed approach; it is very conservative in proposing normalization which is desirable as a preprocessing step for NLP applications. This limitation can be marginalized by providing more data for generating the lexicon. Finally, Example 4 shows also that the system normalize "gr8" which is mainly due to having a flexible similarity cost during the normalization lexicon construction.

<b>1. Source:</b> <i>Mad abt dt so mch intesting</i> <b>Baseline1:</b> <i>Mad at do so much ingesting</i> <b>RW3:</b> <i>Mad about that so much interesting</i>
<b>2. Source:</b> <i>i'l do cuz ma parnts r ma lyf</i> <b>Baseline1:</b> <i>I'll do cut ma parents r ma life</i> <b>RW3:</b> <i>I'll do because my parents are my life</i>
<b>3. Source:</b> <i>yur cuuuuute</i> <b>Baseline1:</b> <i>your cuuuuute</i> <b>RW3:</b> <i>your cute</i>
<b>4. Source:</b> <i>I'm dure u will get a gr8 score</i> <b>Baseline1:</b> <i>I'm dare you will get a gr8 score</i> <b>RW3:</b> <i>I'm dure you will get a great score</i>

Table 5: Normalization Examples

## 5.6 Machine Translation Task Evaluation

The final evaluation of the text normalization system is an extrinsic evaluation where we evaluate the effect of the text normalization task on a social media text translating from English to Spanish using a large scale translation system trained on general domain data. The system is trained on English-Spanish parallel data from WMT 2012 evaluation<sup>4</sup>. The data consists of about 5M parallel sentences on news, europal and UN data. The system is a state of the art phrase based system similar to Moses (Hoang et al., 2007). We used The BLEU score (Papineni et al., 2002) to evaluate the translation accuracy with and without the normalization. Table(6) shows the translation evaluation with different systems. The translation with normalization was improved by about 6% from 29.02 to 30.87 using RW3 as a preprocessing step.

System	BLEU	Impreovemnet
No Normalization	29.02	0%
Baseline1	29.13	0.37%
HB-Dict	29.76	3.69%
RW3	30.87	6.37%

Table 6: Translation Results

## 6 Conclusion and Future Work

We introduced a social media text normalization system that can be deployed as a preprocessor for MT and various NLP applications to handle social media text. The proposed approach is very scalable, adaptive to any domain and language. We show that the proposed unsupervised approach provides a normalization system with very high precision and a reasonable recall. We compared the system with conventional correction approaches and with recent previous work; and we showed that it highly outperforms other systems. Finally, we have used the system as a preprocessing step for a machine translation system which improved the translation quality by 6%.

As an extension to this work, we will extend the approach to handle many-to-many normalization pairs; also we plan to apply the approach to more languages. Furthermore, the approach can be easily extended to handle similar problems such as accent restoration and generic entity normalization.

<sup>4</sup><http://www.statmt.org/wmt12>

## Acknowledgments

We would like to thank Lee Schwartz and Will Lewis for their help in constructing the test sets and in the error analysis. We would also like to thank the anonymous reviewers for their helpful and constructive comments.

## References

- AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pages 3340, Sydney, Australia.
- Eric Brill and Robert C. Moore. 2000. *An improved error model for noisy channel spelling correction*. In ACL 2000: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Englewood Cliffs, NJ, USA.
- Ye-In Chang and Jiun-Rung Chen and Min-Tze Hsu. 2010. A hash trie filter method for approximate string matching in genomic databases. *Applied Intelligence*, 33:1, pages 21:38, Springer US.
- Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal of Document Analysis and Recognition*, vol. 10, pp. 157:174.
- Danish Contractor and Tanveer Faruque and Venkata Subramaniam. 2010. Unsupervised cleansing of noisy text. In COLING '10 Proceedings of the 23rd International Conference on Computational Linguistics, pages 189:196.
- Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text message normalization.. In CALC 09: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, pages 71:78, Boulder, USA.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 600:609, Portland, Oregon.
- Stephan Gouws, Dirk Hovy, and Donald Metzler. 2011. Unsupervised mining of lexical variants from noisy text. In Proceedings of the First workshop on Unsupervised Learning in NLP, pages 82:90, Edinburgh, Scotland.
- Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a twitter. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011), pages 368:378, Portland, Oregon, USA.
- Bo Han and Paul Cook and Timothy Baldwin. 2012. Automatically Constructing a Normalisation Dictionary for Microblogs. Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012), pages 421:432, Jeju Island, Korea.
- Hieu Hoang and Alexandra Birch and Chris Callison-Burch and Richard Zens and Rwth Aachen and Alexandra Constantin and Marcello Federico and Nicola Bertoldi and Chris Dyer and Brooke Cowan and Wade Shen and Christine Moran and Ondrej Bojar. 2007. Moses: Open source toolkit for statistical machine translation.
- Thad Hughes and Daniel Ramage. 2007. Lexical semantic relatedness with random graph walks. Proceedings of Conference on Empirical Methods in Natural Language Processing EMNLP, pp. 581:589, Prague.
- Fei Liu and Fuliang Weng and Bingqing Wang and Yang Liu. 2011. Insertion, Deletion, or Substitution? Normalizing Text Messages without Pre-categorization nor Supervision. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 19:24, Portland, Oregon.
- Dan Melamed. 1999. Bitext Maps and Alignment via Pattern Recognition. In Computational Linguistics, 25, pages 107:130.
- Einat Minkov and William Cohen. Graph Based Similarity Measures for Synonym Extraction from Parsed Text. In Proceedings of the TextGraphs workshop 2012.
- J. Norris. 1997. Markov Chains. Cambridge University Press.
- Kishore Papineni and Salim Roukos and Todd Ward and Wei-jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In Proceedings of ACL-2002: 40th Annual meeting of the Association for Computational Linguistics. , pages 311:318.
- Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. Normalization of non-standard words. 2001.
- Xu Sun and Jianfeng Gao and Daniel Micol and Chris Quirk. 2010. Learning Phrase-Based Spelling Error Models from Clickthrough Data. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 266:274, Sweden.
- Martin Szummer and Tommi. 2002. Partially labeled classification with markov random walks. In Advances in Neural Information Processing Systems, pages 945:952.

Kristina Toutanova and Robert C. Moore. Pronunciation modeling for improved spelling correction.. 2002. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL , pages 144:151, Philadelphia, USA.

Justin Zobel and Philip Dart 1996. Phonetic string matching: Lessons from information retrieval. in Proceedings of the Eighteenth ACM SIGIR International Conference on Research and Development in Information Retrieval, pages 166:173, Zurich, Switzerland.

# Integrating Phrase-based Reordering Features into a Chart-based Decoder for Machine Translation

**ThuyLinh Nguyen**

Language Technologies Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213, USA  
thuylinh@cs.cmu.edu

**Stephan Vogel**

Qatar Computing Research Institute  
Tornado Tower  
Doha, Qatar  
svogel@qf.org.qa

## Abstract

Hiero translation models have two limitations compared to phrase-based models: 1) Limited hypothesis space; 2) No lexicalized reordering model. We propose an extension of Hiero called Phrasal-Hiero to address Hiero's second problem. Phrasal-Hiero still has the same hypothesis space as the original Hiero but incorporates a phrase-based distance cost feature and lexicalized reordering features into the chart decoder. The work consists of two parts: 1) for each Hiero translation derivation, find its corresponding discontinuous phrase-based path. 2) Extend the chart decoder to incorporate features from the phrase-based path. We achieve significant improvement over both Hiero and phrase-based baselines for Arabic-English, Chinese-English and German-English translation.

## 1 Introduction

Phrase-based and tree-based translation model are the two main streams in state-of-the-art machine translation. The tree-based translation model, by using a synchronous context-free grammar formalism, can capture longer reordering between source and target language. Yet, tree-based translation often underperforms phrase-based translation in language pairs with short range reordering such as Arabic-English translation (Zollmann et al., 2008; Birch et al., 2009).

We follow Koehn et al. (2003) for our phrase-based system and Chiang (2005) for our Hiero system. In both systems, the translation of a source sentence  $\mathbf{f}$  is the target sentence  $\mathbf{e}^*$  that maximizes a linear combination of features and weights:

$$\langle \mathbf{e}^*, \mathbf{a}^* \rangle = \operatorname{argmax}_{(\mathbf{e}, \mathbf{a}) \in \mathcal{H}(\mathbf{f})} \sum_{m \in M} \lambda_m h_m(\mathbf{e}, \mathbf{f}, \mathbf{a}). \quad (1)$$

where

- $\mathbf{a}$  is a translation path of  $\mathbf{f}$ . In the phrase-based system,  $\mathbf{a}_{\text{ph}}$  represents a segmentation of  $\mathbf{e}$  and  $\mathbf{f}$  and a correspondance of phrases. In the Hiero system,  $\mathbf{a}_{\text{tr}}$  is a derivation of a parallel parse tree of  $\mathbf{f}$  and  $\mathbf{e}$ , each nonterminal representing a rule in the derivation.
- $\mathcal{H}(\mathbf{f})$  is the hypothesis space of the sentence  $\mathbf{f}$ . We denote  $\mathcal{H}_{\text{ph}}(\mathbf{f})$  as the phrase-based hypothesis space of  $\mathbf{f}$  and  $\mathcal{H}_{\text{tr}}(\mathbf{f})$  as its tree-based hypothesis space. Galley and Manning (2010) point out that due to the hard constraints of rule combination, the tree-based system does not have the same excessive hypothesis space as the phrase-based system.
- $M$  is the set of feature indexes used in the decoder. Many features are shared between phrase-based and tree-based systems including language model, word count, and translation model features. Phrase-based systems often use a lexical reordering model in addition to the distance cost feature.

The biggest difference in a Hiero system and a phrase-based system is in how the reordering is modeled. In the Hiero system, the reordering decision is encoded in weighted translation rules, determined by nonterminal mappings. For example, the rule  $X \rightarrow ne X_1 pas ; not X_1 : w$  indicates the translation of the phrase between *ne* and *pas* to be after the English word *not* with score  $w$ . During decoding, the system parses the source sentence and synchronously generates the target output.

To achieve reordering, the phrase-based system translates source phrases out of order. A reordering distance limit is imposed to avoid search space explosion. Most phrase-based systems are equipped with a distance reordering cost feature to tune the system towards the right amount of reordering, but then also a lexicalized reordering

model to model the direction of adjacent source phrases reordering as either *monotone*, *swap* or *discontinuous*.

There are two reasons to explain the shortcomings of the current Hiero system:

1. A limited hypothesis space because the synchronous context-free grammar is not applicable to non-projective dependencies.
2. It does not have the expressive lexicalized reordering model and distance cost features of the phrase-based system.

When comparing phrase-based and Hiero translation models, most of previous work on tree-based translation addresses its limited hypothesis space problem. Huck et al. (2012) add new rules into the Hiero system, Carreras and Collins (2009) apply the tree adjoining grammar formalism to allow highly flexible reordering. On the other hand, the Hiero model has the advantage of capturing long distance and structure reordering. Galley and Manning (2010) extend phrase-based translation by allowing gaps within phrases such as  $\langle ne \dots pas, not \rangle$ , so the decoder still has the discriminative reordering features of phrase-based, but also uses on average longer phrases. However, these phrase pairs with gaps do not capture structure reordering as do Hiero rules with non-terminal mappings. For example, the rule  $X \rightarrow ne X_1 pas ; not X_1$  explicitly places the translation of the phrase between *ne* and *pas* behind the English word *not* through nonterminal  $X_1$ . This is important for language pairs with strict reordering. In our Chinese-English experiment, the Hiero system still outperforms the discontinuous phrase-based system.

We address the second problem of the original Hiero decoder by mapping Hiero translation derivations to corresponding phrase-based paths, which not only have the same output but also preserve structure distortion of the Hiero translation. We then include phrase-based features into the Hiero decoder.

A phrase-based translation path is the sequence of phrase-pairs, whose source sides cover the source sentence and whose target sides generate the target sentence from left to right. If we look at the leaves of a Hiero derivation tree, the lexicals also form a segmentation of the source and target sentence, thus also form a discontinuous phrase-based translation path. As an example, let us look

at the translation of the French sentence *je ne parle pas le française* into English *i don't speak french* in Figure 1. The Hiero decoder translates the sentence using a derivation of three rules:

- $r_1 = X \rightarrow parle ; speak.$
- $r_2 = X \rightarrow ne X_1 pas ; don't X_1.$
- $r_3 = X \rightarrow Je X_1 le Français ; I X_1 french.$

From this Hiero derivation, we have a segmentation of the sentence pairs into phrase pairs according to the word alignments, as shown on the left side of Figure 1. Ordering these phrase pairs according the word sequence on the target side, shown on the right side of Figure 1, we have a phrase-based translation path consisting of four phrase pairs:  $(je, i)$ ,  $(ne \dots pas, not)$ ,  $(parle, speak)$ ,  $(le française, french)$  that has the same output as the Hiero system. Note that even though the Hiero decoder uses a composition of three rules, the corresponding phrase-based path consists of four phrase pairs. We name this new variant of the Hiero decoder, which uses phrase-based features, Phrasal-Hiero.

Our Phrasal-Hiero addresses the shortcoming of the original Hiero system by incorporating phrase-based features. Let us revisit machine translation's loglinear model combination of features in equation 1. We denote  $ph(\mathbf{a})$  as the corresponding phrase-based path of a Hiero derivation  $\mathbf{a}$ , and  $M_{Ph \setminus H}$  as the indexes of phrase-based features currently not applicable to the Hiero decoder. Our Phrasal-Hiero decoder seeks to find the translation, which optimizes:

$$\langle \mathbf{e}^*, \mathbf{a}^* \rangle = \operatorname{argmax}_{(\mathbf{e}, \mathbf{a}) \in \mathcal{H}_{tr}(\mathbf{f})} \left( \sum_{m \in M_H} \lambda_m h_m(\mathbf{e}, \mathbf{f}, \mathbf{a}) + \sum_{m' \in M_{Ph \setminus H}} \lambda_{m'} h_{m'}(\mathbf{e}, \mathbf{f}, ph(\mathbf{a})) \right).$$

We focus on improving the modelling of reordering within Hiero and include discriminative reordering features (Tillmann, 2004) and a distance cost feature, both of which are not modeled in the original Hiero system. Chiang et al. (2008) added structure distortion features into their decoder and showed improvements in their Chinese-English experiment. To our knowledge, Phrasal-Hiero is the first system, which directly integrates phrase-based and Hiero features into one model.

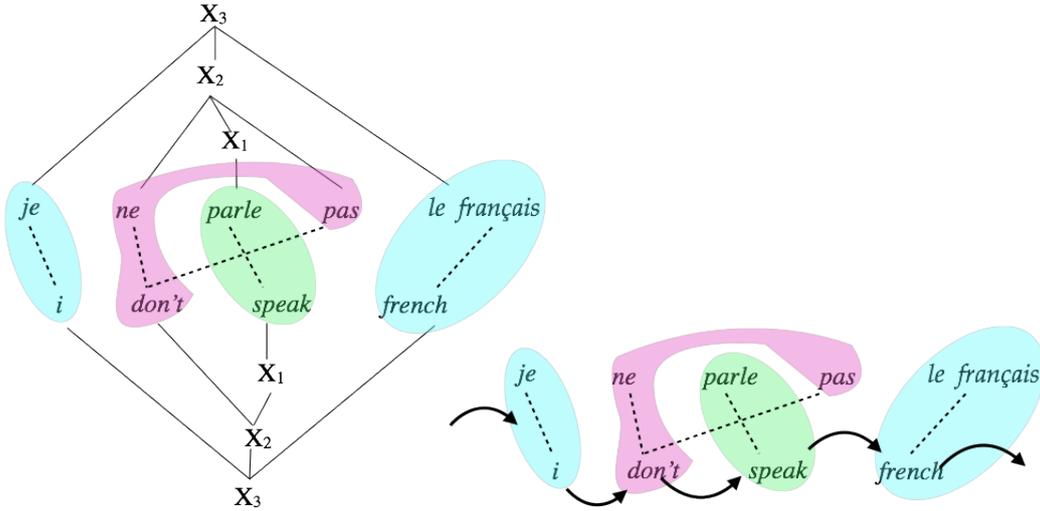


Figure 1: Example of French-English Hiero Translation on the left and its corresponding discontinuous phrase-based translation on the right.

Rules	Alignments	Phrase pairs & nonterminals
$r_1 = X \rightarrow \text{parle} ; \text{ speak}.$	0-0	$(\text{parle} ; \text{ speak})$
$r_2 = X \rightarrow \text{ne } X_1 \text{ pas} ; \text{ don't } X_1.$	0-0 1-1 2-0	$(\text{ne} \dots \text{pas} ; \text{ don't}) ; X_1$
$r_3 = X \rightarrow \text{Je } X_1 \text{ le Francais} ; \text{ I } X_1 \text{ French}$	0-0 1-1 3-2	$(\text{Je} ; \text{ I}) ; X_1 ; (\text{le Francais} ; \text{ french})$
$r_4 = X \rightarrow \text{je } X_1 \text{ le } X_2 ; \text{ i } X_1 X_2$	0-0 1-1 3-2	Not Applicable

Table 1: Rules and their sequences of phrase pairs and nonterminals

Previous work has attempted to weaken the context free assumption of the synchronous context free grammar formalism, for example using syntactic non-terminals (Zollmann and Venugopal, 2006). Our approach can be viewed as applying soft context constraint to make the probability of substituting a nonterminal by a subtree depending on the corresponding phrase-based reordering features.

In the next section, we explain the model in detail.

## 2 Phrasal-Hiero Model

Phrasal-Hiero maps a Hiero derivation into a discontinuous phrase-based translation path by the following two steps:

1. Training: Represent each rule as a sequence of phrase pairs and nonterminals.
2. Decoding: Use the rules' sequences of phrase pairs and nonterminals to find the corresponding phrase-based path of a Hiero derivation and calculate its feature scores.

### 2.1 Map Rule to A Sequence of Phrase Pairs and Nonterminals

We segment the rules' lexical items into phrase pairs. These phrase pairs will be part of the phrase-based translation path in the decoding step. The rules' nonterminals are also preserved in the sequence, during the decoding they will be substituted by other rules' phrase pairs. We now explain how to map a rule to a sequence of phrase pairs and nonterminals.

Let  $r = X \rightarrow s_0 X_1 s_1 \dots X_k s_k ; t_0 X_{\alpha(1)} t_1 \dots X_{\alpha(k)} t_k$  be a rule of  $k$  nonterminals,  $\alpha(\cdot)$  defines the sequence of nonterminals on the target.  $s_i$  or  $t_i$ ,  $i = 0 \dots k$  are phrases between nonterminals, they can be empty because nonterminals can be at the border of the rule or two nonterminals are adjacent. For example the rule  $X \rightarrow \text{ne } X_1 \text{ pas} ; \text{ not } X_1$  has  $k = 1$ ,  $s_0 = \text{ne}$ ,  $s_1 = \text{pas}$ ,  $t_0 = \text{not}$ ,  $t_1$  is an empty phrase because the target  $X_1$  is at the rightmost position.

Phrasal-Hiero retains both nonterminals and lexical alignments of Hiero rules instead of only nonterminal mappings as in (Chiang, 2005). A



erates  $(je, i)$  once for the sentence. Phrase pairs are generated together with phrase-based reordering orientations to build lexicalized reordering table.

### 3 Decoding

Chiang (2007) applied bottom up chart parsing to parse the source sentence and project on the target side for the best translation. Each chart cell  $[X, i, j, r]$  indicates a subtree with rule  $r$  at the root covers the translation of the  $i$ -th word upto the  $j$ -th word of the source sentence. We extend the chart parsing, mapping the subtree to the equivalent discontinuous phrase-based path and includes phrase-based features to the log-linear model.

In Phrasal-Hiero, each chart cell  $[X, i, j, r]$  also stores the first phrase pair and the last phrase pair of the phrase-based translation path covered the  $i$ -th to the  $j$ -th word of the source sentence. These two phrase pairs are the back pointers to calculate reordering features of later larger spans. Because the distance cost feature and phrase-based discriminative reordering feature calculation are both only required the source coverage of two adjacent phrase pairs, we explain here the distance cost calculation.

We will again use three rules  $r_1, r_2, r_3$  in Table 1 and the translation *je ne parle pas le français* into *I don't speak French* to present the technique. Table 3 shows the distance cost calculation.

First, when the rule  $r$  has only terminals, the rule's sequence of phrase pairs and nonterminals consists of only a phrase pair. No calculation is needed, the first phrase pair and the last phrase pair are the same. The chart cell  $X_1 : 2 \dots 2$  in Table 3 shows the translation with the rule  $r_1 = X \rightarrow \textit{parle} ; \textit{speak}$ . The first phrase pair and the last phrase pair point to the phrase  $(\textit{parle}, \textit{speak})$  spanning  $2 \dots 2$  of the source sentence.

When the translation rule's right hand side has nonterminals, the nonterminals in the sequence belong to smaller chart cells that we already found phrase-based paths and calculated their features before. The decoder then substitute these paths into the rule's sequence of phrase pairs and nonterminals to form the complete path for the current span.

We now demonstrate finding the phrase based path and calculate distance cost of the chart cell  $X_2$  spanning  $1 \dots 3$ . The next phrase pair of  $(\textit{ne} \dots \textit{pas}, \textit{don't})$  is the first phrase pair

of the chart cell  $X_1$  which is  $(\textit{parle}, \textit{speak})$ . The distance cost of these two phrase pairs according to discontinuous phrase-based model is  $|2 - 3 - 1| = 2$ . The distance cost of the whole chart cell  $X_2$  also includes the cost of the translation path covered by chart cell  $X_1$  which is 0, therefore the distance cost for  $X_2$  is  $2 + \text{dist}(X_1) = 2$ . We then update the first phrase pair and the last phrase pair of cell  $X_2$ . The first phrase pair of  $X_2$  is  $(\textit{ne} \dots \textit{pas}, \textit{don't})$ , the last phrase pair is also the last phrase pair of cell  $X_1$  which is  $(\textit{parle}, \textit{speak})$ .

Similarly, finding the phrase-based path and calculate its distortion features in the chart cell  $X_3$  include calculate the feature values for moving from the phrase pair  $(\textit{je}, I)$  to the first phrase pair of chart cell  $X_2$  and also from last phrase pair of chart cell  $X_2$  to the phrase pair  $(\textit{le fran}\c{c}aise, \textit{french})$ .

## 4 Experiment Results

In all experiments we use phrase-orientation lexicalized reordering (Galley and Manning, 2008)<sup>2</sup> which models monotone, swap, discontinuous orientations from both reordering with previous phrase pair and with the next phrase pair. There are total six features in lexicalized reordering model.

We will report the impact of integrating phrase-based features into Hiero systems for three language pairs: Arabic-English, Chinese-English and German-English.

### 4.1 System Setup

We are using the following three baselines:

- Phrase-based without lexicalized reordering features. (PB+nolex)
- Phrase-based with lexicalized reordering features.(PB+lex)
- Hiero system with all rules extracted from training data. (Hiero)

We use Moses phrase-based and chart decoder (Koehn et al., 2007) for the baselines. The score difference between PB+nolex and PB+lex results indicates the impact of lexicalized reordering features on phrase-based system. In Phrasal-Hiero we

<sup>2</sup>Galley and Manning (2008) introduce three orientation models for lexicalized reordering: word-based, phrase-based and hierarchical orientation model. We apply phrase-based orientation in all experiment using lexicalized reordering.

Chart Cell	Rule’s phrase pairs & NTs	Distance	First Phrase Pair	Last Phrase Pair
$X_1 : 2 \dots 2$	$(parle, speak)$	$\emptyset$	$2 \dots 2 (parle, speak)$	
$X_2 : 1 \dots 3$	$(ne \dots pas, don't) ; X_1$	$2 + \text{dist}(X_1) = 2$	$1 \dots 3 (ne \dots pas, don't)$	$2 \dots 2 (parle, speak)$
$X_3 : 0 \dots 5$	$(Je ; I) ; X_2 ; (le Franais; french)$	$0 + \text{dist}(X_2) + 1 = 3$	$0 \dots 0 (je, I)$	$4 \dots 5 (le Franais; french)$

Table 3: Phrasal-Hiero Decoding Example: Calculate distance cost feature for the translation in Figure 1.

will compare if these improvements still carry on into Hiero systems.

The original Hiero system with all rules extracted from training data (Hiero) is the most relevant baseline. We will evaluate the difference between this Hiero baseline and our Phrasal-Hiero.

To implement Phrasal-Hiero, we extended Moses chart decoder (Koehn et al., 2007) to include distance-based reordering as well as the lexicalized phrase orientation reordering model. We will report the following results for Phrasal-Hiero:

- Hiero translation results on the subset of rules without unaligned phrases. (we denote this in the table scores as P.H.)
- Phrasal-Hiero with phrase-based distance cost feature (P.H.+dist).
- Phrasal-Hiero with phrase-based lexicalized reordering features(P.H.+lex).
- Phrasal-Hiero with distance cost and lexicalized reordering features(P.H.+dist+lex).

## 4.2 Arabic-English Results

The Arabic-English system was trained from 264K sentence pairs with true case English. The Arabic is in ATB morphology format. The language model is the interpolation of 5-gram language models built from news corpora of the NIST 2012 evaluation. We tuned the parameters on the MT06 NIST test set (1664 sentences) and report the BLEU scores on three unseen test sets: MT04 (1353 sentences), MT05 (1056 sentences) and MT09 (1313 sentences). All test sets have four references per each sentence.

The results are in Table 4. The three rows in the first block are the baseline scores. Phrase-based with lexicalized reordering features(PB+lex) shows significant improvement on all test sets over the simple phrase-based system without lexicalized reordering (PB+nolex). On average the improvement is 1.07 BLEU score (45.66

	MT04	MT05	MT09	Avg.
PB+nolex	47.40	46.83	42.75	45.66
PB+lex	48.62	48.07	43.51	46.73
Hiero	48.17	47.85	42.37	46.13
P.H. (48.54% rules)	48.52	47.78	42.80	46.37
P.H.+dist	48.46	47.92	42.62	46.33
P.H. +lex	48.70	48.59	<b>43.84</b>	47.04
P.H +lex+dist	<b>49.35</b>	<b>49.07</b>	43.40	<b>47.27</b>
Improv. over PB+lex	0.73	1.00	0.34	0.54
Improv. over P.H.	0.83	1.29	1.04	0.90
Improv. over Hiero	1.18	1.22	1.47	1.14

Table 4: Arabic-English true case translation scores in BLEU metric. The three rows in the first block are the baseline scores. The next four rows in the second block are Phrasal-Hiero scores, the best scores are in boldface. The three rows in the last block are the Phrasal-Hiero improvements.

versus 46.73). We make the same observation as Zollmann et al. (2008), i.e., that the Hiero baseline system underperforms compared to the phrase-based system with lexicalized phrase-based reordering for Arabic-English in all test sets, on average by about 0.60 BLEU points (46.13 versus 46.73). This is because Arabic language has relative free reordering, but mostly short distance, which is better captured by discriminative reordering features.

The next four rows in the second block of Table 4 show Phrasal-Hiero results. The P.H. line is the result of Hiero experiment on only a subset of rules without nonaligned phrases. As mentioned in section 2.1, Phrasal-Hiero only uses 48.54% of the rules but achieves as good or even better performance (on average 0.24 BLEU points better) compared to the original Hiero system using the full set of rules.

We do not benefit from adding only the

distance-based reordering feature (P.H+dist) to the Arabic-English experiment but get significant improvements when adding the six features of the lexicalized reordering (P.H+lex). Table 4 shows that the P.H.+lex system gains on average 0.67 BLEU points (47.04 versus 46.37). Even though the baseline Hiero underperforms phrase-based system with lexicalized reordering(P.B+lex), the P.H.+lex system already outperforms P.B+lex in all test sets (on average 47.04 versus 46.73).

Adding both distance cost and lexicalized reordering features (P.H.+dist+lex) performs the best. On average P.H.+dist+lex improves 0.90 BLEU points over P.H. without new phrase-based features and 1.14 BLEU score over the baseline Hiero system. Note that Hiero rules already have lexical context in the reordering, but adding phrase-based lexicalized reordering features to the system still gives us about as much improvement as the phrase-based system gets from lexicalized reordering features, here 1.07 BLEU points. And our best Phrasal-Hiero significantly improves over the best phrase-based baseline by 0.54 BLEU points. This shows that the underperformance of the Hiero system is due to its lack of lexicalized reordering features rather than a limited hypothesis space.

### 4.3 Chinese-English Results

The Chinese-English system was trained on FBIS corpora of 384K sentence pairs, the English corpus is lower case. The language model is the trigram SRI language model built from Xinhua corpus of 180 millions words. We tuned the parameters on MT06 NIST test set of 1664 sentences and report the results of MT04, MT05 and MT08 unseen test sets. The results are in Table 5.

We also make the same observation as Zollmann et al. (2008) on the baselines for Chinese-English translation. Even though the phrase-based system benefits from lexicalized reordering, PB+lex on average outperforms PB+nolex by 1.16 BLEU points (25.87 versus 27.03), it is the Hiero system that has the best baseline scores across all test sets, with an average of 27.70 BLEU points.

Phrasal Hiero scores are given in the second block of Table 5. It uses 84.19% of the total training rules, but unlike the Arabic-English system, using a subset of the rules costs Phrasal-Hiero on all test sets and on average it loses 0.49 BLEU points (27.21 versus 27.70). Similar to Chiang

	MT04	MT05	MT08	Avg.
PB+nolex	29.99	26.4	21.23	25.87
PB+lex	31.03	27.57	22.41	27.03
Hiero	32.49	28.06	22.57	27.70
P.H. (84.19% rules)	31.83	27.66	22.16	27.21
P.H.+dist	32.18	28.25	22.46	27.63
P.H.+lex	32.55	28.51	23.08	28.05
P.H.+lex+dist	<b>33.06</b>	<b>28.78</b>	<b>23.23</b>	<b>28.35</b>
Improv. over PB+lex	2.03	1.21	0.82	1.32
Improv. over P.H.	1.23	1.12	1.07	1.14
Improv. over Hiero	0.57	0.72	0.66	0.65

Table 5: Chinese-English lower case translation scores in BLEU metric.

et al. (2008) in their Chinese-English experiment, we benefit by adding the distance cost feature. PH.+dist outperforms P.H. on all test sets. We have better improvements when adding the six features of the lexicalized reordering model: P.H.+lex on average has 28.05 BLEU points, i.e. gains 0.84 over P.H.. The P.H.+lex system is even better than the best Hiero baseline using the whole set of rules.

We again get the best translation when adding both the distance cost feature and the lexicalized reordering features. The P.H.+dist+lex has the best score across all the test sets and on average gains 1.14 BLEU points over P.H. So adding phrase-based features to the Hiero system yields nearly the same improvement as adding lexicalized reordering features to the phrase-based system. This shows that a strong Chinese-English Hiero system still benefits from phrase-based features. Furthermore, the P.H.+dist+lex also outperforms the Hiero baseline using all rules from training data.

### 4.4 German-English Results

We next consider German-English translation. The systems were trained on 1.8 million sentence pairs using the Europarl corpora. The language model is three-gram SRILM trained from the target side of the training corpora. We use WMT 2010 (2489 sentences) as development set and report scores on WMT 2008 (2051 sentences), WMT 2009 (2525 sentences), WMT 2011 (3003 sentences). All test sets have one reference per test sentence. The results are in Table 6.

WMT test	08	09	11	Avg.
PB+nolex	17.46	17.38	16.76	17.20
PB+lex	18.16	17.85	17.18	17.73
Hiero	18.20	18.23	17.46	17.96
P.H. (80.54% rules)	18.24	18.15	17.39	17.92
P.H. +dist	18.19	17.97	17.41	17.85
P.H. +lex	18.59	18.46	17.69	18.24
P.H.+lex+dist	<b>18.70</b>	<b>18.53</b>	<b>17.81</b>	<b>18.34</b>
Improv. over PB+lex	0.54	0.68	0.63	0.61
Improv. over P.H.	0.46	0.38	0.42	0.42
Improv. over Hiero	0.50	0.30	0.35	0.38

Table 6: German-English lower case translation scores in BLEU metric.

The Hiero baseline performs on average 0.26 BLEU points better than the phrase-based system with lexicalized reordering features (PB+lex). The Phrasal-Hiero system used 80.54% of the total training rules, but on average the P.H. system has the same performance as the Hiero system using all the rules extracted from training data. Similar to the Arabic-English experiment, Phrasal-Hiero does not benefit from adding the distance cost feature. We do, however, see improvements on all test sets when adding lexicalized reordering features. On average the P.H.+lex results are 0.32 BLEU points higher than the P.H. results. The best scores are achieved with P.H.+lex+dist. The German-English translations on average gain 0.38 BLEU score by adding both distance cost and discriminative reordering features.

#### 4.5 Impact of segment rules into phrase pairs

Phrasal Hiero is the first system using rules' lexical alignments. If lexical alignments are not available, we can not divide the rules' lexicals into phrase pairs without losing their dependancies. An alternative approach would be combining all lexicals of a rule into one phrase pair. We run an addition experiment for this approach on Arabic-English dataset. Table 7 shows the examples rules and its new sequence of nonterminals and phrase pairs. The rules  $r_1$  and  $r_2$  have the same sequences as in Table 1. Without segment rules into phrase pairs, the rule  $r_3$  has only one phrase pair:  $ph = (Je...le Francaise ; I...french)$  and

$ph$  is repeated twice in  $r_3$ 's sequence of phrase pairs and nonterminals. The new experiment uses the complete set of rules so the rule  $r_4$  is included.

According to the new sequence of phrase pairs and nonterminals, during decoding the rule  $r_3$  has *discontinuous* translation directions on both from phrase pair  $ph$  to the nonterminal  $X_1$  and from  $X_1$  to  $ph$ . But using lexical alignment and divide the rule into phrase pairs as in section 2.1, the sequence preserves the translation order of  $r_3$  as two *monotone* translations from  $(je; I)$  to  $X_1$  and from  $X_1$  to  $(le Francaise ; french)$ .

	Avg
Hiero	46.13
Hiero+lex (no lex. alignments)	46.45 (+0.32)
P.H.	46.37
P.H.+lex (with lex. alignments)	47.04 (+0.67)

Table 8: Average of Arabic-English translation scores in BLEU metric. Compare the improvement of using rules' lexical alignments (2nd block) and not using rules' lexical alignments (1st block).

Table 8 compares the two experiments results. The additional experiment is denoted as Hiero+lex in the table. The first block shows an improvement of 0.32 BLEU score when adding discriminated reordering features on Hiero (using the whole set of rules and no rule segmentation). The second block is the impact of adding discriminated reordering features on Phrasal Hiero (using a subset of rules and segment rules into phrase pairs). Here the improvement of P.H.+lex over P.H. is 0.67 BLEU score. It shows the benefit of segment rules into phrase pairs.

#### 4.6 Rules without unaligned phrases

	A-E	C-E	G-E
Hiero	46.13	27.70	17.96
P.H.	46.36	27.21	17.92
%Rules used	48.54%	84.19%	80.54%
P.H.+lex+dist	47.27	28.35	18.34

Table 9: The impact of using only rules without nonaligned phrases on Phrasal-Hiero results.

Table 9 summarizes the impact of using only rules without nonaligned phrases on Phrasal-

Rules	Phrase pairs & nonterminals
$r_1 = X \rightarrow \text{parle} ; \text{ speak}.$	$(\text{parle} ; \text{ speak})$
$r_2 = X \rightarrow \text{ne } X_1 \text{ pas} ; \text{ don't } X_1.$	$(\text{ne} \dots \text{pas} ; \text{ don't}) ; X_1$
$r_3 = X \rightarrow \text{Je } X_1 \text{ le Francais} ; \text{ I } X_1 \text{ French}$	$(\text{Je} \dots \text{le Francais} ; \text{ I} \dots \text{ french}) ; X_1 ;$ $(\text{Je} \dots \text{le Francais} ; \text{ I} \dots \text{ french})$
$r_4 = X \rightarrow \text{je } X_1 \text{ le } X_2 ; \text{ i } X_1 X_2$	$(\text{je} \dots \text{le} ; \text{ i}) ; X_1 ; X_2$

Table 7: Example of translation rules and their sequences of phrase pairs and nonterminals when lexical alignments are not available.

Hiero. Using only rules without nonaligned phrases can get the same performance with translation with full set of rules for Arabic-English and German-English experiments but underperforms for the Chinese-English system. We suggest the difference might come from the linguistic divergences of source and target languages.

Phrasal Hiero includes all lexical rules (rules without nonterminal) therefore it still has the same lexical coverage as the original Hiero system. In the Arabic-English system, the Arabic is in ATB format, therefore most English words should have alignments in the ATB source, rules with nonaligned phrases could be the results of bad alignments or non-informative rules, therefore we could have better performance by using a subset of rules in Phrasal-Hiero.

As Chinese and English are highly divergent, we expect many phrases in one language correctly unaligned in the other language. So leaving out the rules with nonaligned phrases could degrade the system. Even though the current Phrasal-Hiero with extra phrase-based features outperforms the Hiero baseline, future work for Phrasal-Hiero will focus on including all rules extracted from training corpora.

#### 4.7 Discontinuous Phrase-Based

	C-E	G-E
PB+lex	27.03	17.73
PB+lex+gap	27.11	17.55
Hiero	27.70	17.96
P.H.+lex+dist	28.35	18.34

Table 10: Comparing Phrasal-Hiero with translation with gap for Chinese-English and German-English. The numbers are average BLEU scores of all test sets.

We compare Phrasal-Hiero with a discontinuous phrase-based system introduced by Galley and

Manning (2010) for Chinese-English and German-English system. Table 10 shows the average results. We used Phrasal decoder (Cer et al., 2010) for phrase-based with gaps (PB+lex+gap) results. While we do not focus on the differences in the toolkits, our Phrasal-Hiero still outperforms the phrase-based with gaps experiments.

## Conclusion

We have presented a technique to combine phrase-based features and tree-based features into one model. Adding a distance cost feature, we only get better translation for Chinese-English translation. Phrasal-Hiero benefits from adding discriminative reordering features in all experiment. We achieved the best result when adding both distance cost and lexicalized reordering features. Phrasal-Hiero currently uses only a subset of rules from training data. A future work on the model can include complete rule sets together with word insertion/deletion features for nonaligned phrases.

## References

- A. Birch, P. Blunsom, and M. Osborne. 2009. A Quantitative Analysis of Reordering Phenomena. In *StatMT '09: Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 197–205.
- X. Carreras and M. Collins. 2009. Non-Projective Parsing for Statistical Machine Translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 200–209.
- D. Cer, M. Galley, D. Jurafsky, and C. Manning. 2010. Phrasal: A Statistical Machine Translation Toolkit for Exploring New Model Features. In *Proceedings of the NAACL HLT 2010 Demonstration Session*, pages 9–12. Association for Computational Linguistics, June.
- D. Chiang, Y. Marton, and P. Resnik. 2008. Online Large-Margin Training of Syntactic and Structural Translation Features. In *Proceedings of the Conference on Empirical Methods in Natural Language*

- Processing*, pages 224–233. Association for Computational Linguistics.
- D. Chiang. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proc. of ACL*.
- D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- M. Galley and C. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 847–855, Honolulu, Hawaii, October.
- M. Galley and C. D. Manning. 2010. Accurate Non-Hierarchical Phrase-Based Translation. In *Proceedings of NAACL-HLT*, pages 966–974.
- M. Huck, S. Peitz, M. Freitag, and H. Ney. 2012. Discriminative Reordering Extensions for Hierarchical Phrase-Based Machine Translation. In *EAMT*, pages 313–320.
- P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proc. of HLT-NAACL*, pages 127–133.
- P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL demonstration session*.
- C. Tillmann. 2004. A Unigram Orientation Model for Statistical Machine Translation. In *Proceedings of HLT-NAACL: Short Papers*, pages 101–104.
- A. Zollmann and A. Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In *Proc. of NAACL 2006 - Workshop on Statistical Machine Translation*.
- A. Zollmann, A. Venugopal, F. Och, and J. Ponte. 2008. A Systematic Comparison of Phrase-Based, Hierarchical and Syntax-Augmented Statistical MT. In *Proceedings of the Conference on Computational Linguistics (COLING)*.

# Machine Translation Detection from Monolingual Web-Text

**Yuki Arase**

Microsoft Research Asia  
No. 5 Danling St., Haidian Dist.  
Beijing, P.R. China  
yukiar@microsoft.com

**Ming Zhou**

Microsoft Research Asia  
No. 5 Danling St., Haidian Dist.  
Beijing, P.R. China  
mingzhou@microsoft.com

## Abstract

We propose a method for automatically detecting low-quality Web-text translated by statistical machine translation (SMT) systems. We focus on the *phrase salad* phenomenon that is observed in existing SMT results and propose a set of computationally inexpensive features to effectively detect such machine-translated sentences from a large-scale Web-mined text. Unlike previous approaches that require bilingual data, our method uses only *monolingual* text as input; therefore it is applicable for refining data produced by a variety of Web-mining activities. Evaluation results show that the proposed method achieves an accuracy of 95.8% for sentences and 80.6% for text in noisy Web pages.

## 1 Introduction

The Web provides an extremely large volume of textual content on diverse topics and areas. Such data is beneficial for constructing a large scale monolingual (Microsoft Web N-gram Services, 2010; Google N-gram Corpus, 2006) and bilingual (Nie et al., 1999; Shi et al., 2006; Ishisaka et al., 2009; Jiang et al., 2009) corpus that can be used for training statistical models for NLP tools, as well as for building a large-scale knowledge-base (Suchanek et al., 2007; Zhu et al., 2009; Fader et al., 2011; Nakashole et al., 2012). With recent advances in statistical machine translation (SMT) systems and their wide adoption in Web services through APIs (Microsoft Translator, 2009; Google Translate, 2006), a large amount of text in Web pages is translated by SMT systems. According to Rarrick et al. (2011), their Web crawler finds that more than 15% of English-Japanese parallel documents are machine translation. Machine-translated sentences are useful if

they are of sufficient quality and indistinguishable from human-generated sentences; however, the quality of these machine-translated sentences is generally much lower than sentences generated by native speakers and professional translators. Therefore, a method to detect and filter such SMT results is desired to best make use of Web-mined data.

To solve this problem, we propose a method for automatically detecting Web-text translated by SMT systems<sup>1</sup>. We especially target machine-translated text produced through the Web APIs that is rapidly increasing. We focus on the *phrase salad* phenomenon (Lopez, 2008), which characterizes translations by existing SMT systems, *i.e.*, each phrase in a sentence is semantically and syntactically correct but becomes incorrect when combined with other phrases in the sentence. Based on this trait, we propose features for evaluating the likelihood of machine-translated sentences and use a classifier to determine whether the sentence is generated by the SMT systems.

The primary contributions of the proposed method are threefold. First, unlike previous studies that use parallel text and bilingual features, such as (Rarrick et al., 2011), our method only requires monolingual text as input. Therefore, our method can be used in monolingual Web data mining where bilingual information is unavailable. Second, the proposed features are designed to be computationally light so that the method is suitable for handling a large-scale Web-mined data. Our method determines if an input sentence contains phrase salads using a simple yet effective features, *i.e.*, language models (LMs) and automatically obtained non-contiguous phrases that are frequently used by people but difficult for SMT systems to generate. Third, our method computes features using both human-generated text and SMT

<sup>1</sup>In this paper, the term *machine-translated* is used for indicating translation by SMT systems.

results to capture a phrase salad by contrasting these features, which significantly improves detection accuracy.

We evaluate our method using Japanese and English datasets, including a human evaluation to assess its performance. The results show that our method achieves an accuracy of 95.8% for sentences and 80.6% for noisy Web-text.

## 2 Related Work

Previous methods for detecting machine-translated text are mostly designed for bilingual corpus construction. Antonova and Misyurev (2011) design a phrase-based decoder for detecting machine-translated documents in Russian-English Web data. By evaluating the BLEU score (Papineni et al., 2002) of translated documents (by their decoder) against the target-side documents, machine translation (MT) results are detected. Rarrick et al. (2011) extract a variety of features, such as the number of tokens and character types, from sentences in both the source and target languages to capture words that are mis-translated by MT systems. With these features, the likelihood of a bilingual sentence pair being machine-translated can be determined.

Confidence estimation of MT results is also a related area. These studies aim to precisely replicate human judgment in terms of the quality of machine-translated sentences based on features extracted using a syntactic parser (Corston-Oliver et al., 2001; Gamon et al., 2005; Avramidis et al., 2011) or essay scoring system (Parton et al., 2011), assuming that their input is always machine-translated. In contrast, our method aims at making a binary judgment to distinguish machine-translated sentences from a mixture of machine-translated and human-generated sentences. In addition, although methods for confidence estimation can assume sentences of a known source language and reference translations as inputs, these are unavailable in our problem setting.

Another related area is automatic grammatical error detection for English as a second language (ESL) learners (Leacock et al., 2010). We use common features that are also used in this area. They target specific error types commonly made by ESL learners, such as errors in prepositions and subject-verb agreement. In contrast, our method does not specify error types and aims to de-

tect machine-translated sentences focusing on the phrase salad phenomenon produced by SMT systems. In addition, errors generated by ESL learners and SMT systems are different. ESL learners make spelling and grammar mistakes at the word level but their sentence are generally structured while SMT results are unstructured due to phrase salads. Works on *translationese* detection (Baroni and Bernardini, 2005; Kurokawa et al., 2009; Ilisei et al., 2010) aim to automatically identify *human-translated* text by professionals using text generated by native speakers. These are related, but our work focuses on machine-translated text.

The closest to our approach is the method proposed by Moore and Lewis (2010). It automatically selects data for creating a domain-specific LM. Specifically, the method constructs LMs using corpora of target and non-target domains and computes a cross-entropy score of an input sentence for estimating the likelihood that the input sentence belongs to the target or non-target domains. While the context is different, our work uses a similar idea of data selection for the purpose of detecting low-quality sentences translated by SMT systems.

## 3 Proposed Method

When APIs of SMT services are used for machine-translating an Web page, they typically insert specific tags into the HTML source. Utilizing such tags makes MT detection trivial. However, the actual situation is more complicated in real Web data. When people manually copy and paste machine-translated sentences, such tags are lost. In addition, human-generated and machine-translated sentences are often mixed together even in a single paragraph. To observe the distribution of machine-translated sentences in such difficult cases, we examine 3K sentences collected by our in-house Web crawler. Among them, excluding the pages with the tags of MT APIs, 6.7% of them are found to be clearly machine translation. Our goal is to automatically identify these sentences that cannot be simply detected by the tags, except when the sentences are of sufficient quality to be indistinguishable from human-generated sentences.

### 3.1 Phrase Salad Phenomenon

Fig. 1 illustrates the phrase salad phenomenon that characterizes a sentence translated by an existing

| Of surprise | was up | foreigners flocked | overseas | as well, | they publicized not only | Japan, | saw an article from the news. |  
 Natural English: The news was broadcasted not only in Japan but also overseas, and it surprised foreigners who read the article.

Figure 1: The phrase salad phenomenon in a sentence translated by an SMT system; each (segmented) phrase is correct and fluent, but dotted arcs show unnatural sequences of phrases and the boxed phrase shows an incomplete non-contiguous phrase.

SMT system. Each phrase, a sequence of consecutive words, is fluent and grammatically correct; however, the fluency and grammar correctness are both poor in inter-phrases. In addition, a phrase salad becomes obvious by observing distant phrases. For example, the boxed phrase in Fig. 1 is a part of the non-contiguous phrase “not only  $\star$  but also<sup>2</sup>,” however, it lacks the latter part of the phrase (“but also”) that is also necessary for composing a meaning. Such non-contiguous phrases are difficult for most SMT systems to generate, since these phrases require insertion of sub-phrases in distant parts of the sentence.

Based on the observation of these characteristics, we define features to capture a phrase salad by examining local and distant phrases. These features evaluate (1) fluency (Sec. 3.2), (2) grammaticality (Sec. 3.3), and (3) completeness of non-contiguous phrases in a sentence (Sec. 3.4). Furthermore, humans can distinguish machine-translated text because they have prior knowledge of how a human-generated sentence would look like, which has been accumulated by observing a lot of examples through their life. This knowledge makes phrase-salads, e.g., missing objects and influent sequence of words, obvious for humans since they rarely appear on human-generated sentences. Based on this assumption, we extract these features using both human-generated and machine-translated text. Features extracted from human-generated text represent the similarity to human-generated text. Likewise, features extracted from machine-translated text depict the similarity to machine-translated text. By contrasting these feature weights, we can effectively capture phrase salads in the sentence.

### 3.2 Fluency Feature

In a machine-translated sentence, fluency becomes poor among phrases where a phrase salad occurs. We capture this infuency using two independent LM scores;  $f_{w,H}$  and  $f_{w,MT}$ . The former LM is

<sup>2</sup>We use the symbol  $\star$  to represent a gap in which any word or phrase can be placed.

trained with human-generated sentences and the latter one is trained with machine-translated sentences. We input a sentence into both of the LMs and use the scores as the fluency features.

### 3.3 Grammaticality Feature

In a sentence with phrase salads, its grammaticality is poor because tense and voice become inconsistent among phrases. We capture this using LMs trained with part-of-speech (POS) sequences of human-generated and machine-translated sentences, and the features of  $f_{pos,H}$  and  $f_{pos,MT}$  are respectively computed. In a similar manner with a word-based LM, such grammatical inconsistency among phrases is detectable when computing a POS LM score, since the score becomes worse when an  $N$ -gram covers inter-phrases where a phrase salad occurs. This approach achieves computational efficiency since it only requires a POS tagger.

Since a phrase salad may occur among distant phrases of a sentence, it is also effective to evaluate combinations of phrases that cannot be covered by the span of  $N$ -gram. For this purpose, we make use of function words that sparsely appear in a sentence where their combinations are syntactically constrained. For example, the same preposition rarely appears many times in a human-generated sentence, while it does in a machine-translated sentence due to the phrase salad. Similar to the POS LM, we first analyze sentences generated by human or SMT by a POS tagger, extract sequences of function words, and finally train LMs with the sequences. We use these LMs to obtain scores that are used as features  $f_{fw,H}$  and  $f_{fw,MT}$ .

### 3.4 Gappy-Phrase Feature

There are a lot of common non-contiguous phrases that consist of sub-phrases (contiguous word string) and gaps, which we refer to as *gappy-phrases* (Bansal et al., 2011). We specifically use gappy-phrases of 2-tuple, *i.e.*, phrases consisting of two sub-phrases and one gap in the middle. Let us take an English example “not only  $\star$  but

Sequences
World population not only grows , but grows old .
A press release not only informs but also teases .
Hazelnuts are not only for food , but also fuel .
The coalition must not only listen but also act .

Table 1: Example of a sequence database

also.” When a sentence contains the phrase “not only,” the phrase “but also” is likely to appear in human-generated sentences. Such a gappy-phrase is difficult for SMT systems to correctly generate and causes a phrase salad. Therefore, we define a feature to evaluate how likely a sentence contains gappy-phrases in a complete form without missing sub-phrases. This feature is effective to complement LMs that capture characteristics in  $N$ -grams.

**Sequential Pattern Mining** It is costly to manually collect a lot of such gappy-phrases. Therefore, we regard the task as sequential pattern mining and apply PrefixSpan proposed by Pei et al. (2001), which is a widely used sequential pattern mining method<sup>3</sup>.

Given a set of sequences and a user-specified  $min\_support \in \mathbb{N}$  threshold, the sequential pattern mining finds all frequent subsequences whose occurrence frequency is no less than  $min\_support$ . For example, given a sequence database like Table 1, the sequential pattern mining finds all frequent subsequences, e.g., “not only,” “not only  $\star$  but also,” “not  $\star$  but  $\star$ ,” and *etc.*

To capture a phrase salad by contrasting appearance of gappy-phrases in human-generated and machine-translated text, we independently extract gappy-phrases from each of them using PrefixSpan. We then compute features  $f_{g,H}$  and  $f_{g,MT}$  using the obtained phrases.

### Observation of Extracted Gappy-Phrases

Based on a preliminary experiment, we set the parameter  $min\_support$  of PrefixSpan to 100 for computational efficiency. We extract gappy-phrases (of 2-tuple) from our development dataset described in Sec. 4.1 that includes 254K human-generated and 134K machine-translated sentences in Japanese, and 210K human-generated and 159K machine-translated sentences in English.

Regarding the Japanese dataset, we obtain about 104K and 64K gappy-phrases from human-

generated and machine-translated sentences, respectively. According to our observation of the extracted phrases, 21K phrases commonly appear in human-generated and machine-translated sentences. Many of these common phrases are incomplete forms of gappy-phrases that lack semantic meaning to humans, such as “not only  $\star$  the” and “not only  $\star$  and.” On the other hand, complete forms of gappy-phrases that preserve semantic meaning exclusively appear in phrases extracted from human-generated sentences. We also obtain about 74K and 42K phrases from human-generated and machine-translated sentences in the English dataset (21K of them are common).

**Phrase Selection** As a result of sequential pattern mining, we can gather a huge number of gappy-phrases from human-generated and machine-translated text, but as we described above, many of them are common. In addition, it is computationally expensive to use all of them. Therefore, our method selects useful phrases for detecting machine-translated sentences.

Although there are several approaches for feature selection, e.g., (Sebastiani, 2002), we use a method that is suitable for handling a large number of feature candidates. Specifically, we evaluate gappy-phrases based on the information gain that measures the amount of information in bits obtained for class prediction when knowing the presence or absence of a phrase and the corresponding class distribution. This corresponds to measuring an expected reduction in entropy, i.e., uncertainty associated with a random factor. The information gain  $G \in \mathbb{R}$  for a gappy-phrase  $g$  is defined as

$$G(g) \doteq H(C) - P(X_g^1)H(C|X_g^1) - P(X_g^0)H(C|X_g^0),$$

where  $H(C)$  represents the entropy of the classification,  $C$  is a stochastic variable taking a class,  $X_g$  is a stochastic variable representing the presence ( $X_g^1$ ) or absence ( $X_g^0$ ) of the phrase  $g$ ,  $P(X_g)$  represents the probability of presence or absence of the phrase  $g$ , and  $H(C|X_g)$  is the conditional entropy due to the phrase  $g$ . We use top- $k$  phrases based on the information gain  $G$ . Specifically, we use the top 40% of phrases to compute the feature values. Table 2 shows examples of gappy-phrases extracted from human-generated and machine-translated text in our development dataset and remain after feature selection.

<sup>3</sup>Due to the severe space limitation, readers are referred to that paper.

Human	in the early * period	MT	after * after the
	known as * to		and also * and
	more * than		and * but the
	not only * but also		no * not
	with * as well as		not * not

Table 2: Example of gappy-phrases extracted from human-generated and machine-translated text; phrases preserving semantic meaning are extracted only from human-generated text.

The gappy-phrases depend on each other, and the more phrases extracted from human-generated (machine-translated) text are found in a sentence, the more likely the sentence is human-generated (machine-translated). Therefore, we compute the feature as

$$f_c(s) = \sum_{i \in k} w_i \delta(i, s),$$

where  $w_i$  is a weight of the  $i$ -th phrase, and  $\delta(i, s)$  is a Kronecker’s delta function that takes 1 if the sentence  $s$  includes the  $i$ -th phrase and takes 0 otherwise. We may set the weight  $w_i$  according to the importance of the phrase, such as the information gain. In this work, we set  $w_i$  to 1 for simplicity.

### 3.5 Classification

Table 3 summarizes the features employed in our method. In addition to the discussed features, we use the length of a sentence as a feature  $f_{len}$  to avoid the bias of LM-based features that favor shorter sentences. The proposed method takes a monolingual sentence from Web data as input and computes a feature vector of  $f = (f_{w,H}, \dots, f_{len}) \in \mathbb{R}^9$ . Each feature is finally normalized to have a zero-mean and unit variance distribution. In the feature space, a support vector machine (SVM) classifier (Vapnik, 1995) is used to determine the likelihoods of machine-translated and human-generated sentences.

## 4 Experiments

We evaluate our method using both Japanese and English datasets from various aspects and investigate its characteristics. In this section, we describe our experiment settings.

### 4.1 Data Preparation

For the purpose of evaluation, we use human-generated and machine-translated sentences for

Feature	Notation
Fluency	$f_{w,H}, f_{w,MT}$
Grammaticality	$f_{pos,H}, f_{pos,MT}$ $f_{fw,H}, f_{fw,MT}$
Gappy-phrase	$f_{g,H}, f_{g,MT}$
Length	$f_{len}$

Table 3: List of proposed features and their notations

constructing LMs, extracting gappy-phrases, and training a classifier. These sentences should be ensured to be human-generated or machine-translated, and the human-generated and machine-translated sentences express the same content for fairness of evaluation to avoid effects due to vocabulary difference.

As a dataset that meets these requirements, we use parallel text in public websites (this is for fair evaluation and our method can be trained using *nonparallel* text on an actual deployment). Eight popular sites having Japanese and English parallel pages are crawled, whose text is manually verified to be human-generated. The main textual content of these 131K parallel pages are extracted, and the sentences are aligned using (Ma, 2006). As illustrated in Fig. 2, the text in one language is fed to the Bing translator, Google Translate, and an in-house SMT system<sup>4</sup> implemented based on (Chiang, 2005) by ourselves for obtaining sentences translated by SMT systems. Due to a severe limitation on the number of requests to the APIs, we randomly subsample sentences before sending them to these SMT systems. We use text in the other language as human-generated sentences<sup>5</sup>.

In this manner, we prepare 508K human-generated and 268K machine-translated sentences as a Japanese dataset, and 420K human-generated and 318K machine-translated sentences as an English dataset. We split each of them into two even datasets and use one for development and the other for evaluation.

### 4.2 Experiment Setting

For the fluency and grammaticality features, we train 4-gram LMs using the development dataset with the SRI toolkit (Stolcke, 2002). To obtain the POS information, we use Mecab (Kudo et al., 2004) for Japanese and a POS tagger developed by Toutanova et al. (2003) for English. We evaluate

<sup>4</sup>A preliminary evaluation of the in-house SMT system shows that it has comparable quality with Bing translator.

<sup>5</sup>These are a mixture of sentences generated by native speakers and professional translators/editors.

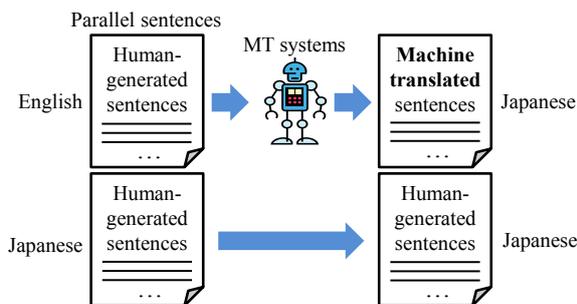


Figure 2: Experimental data preparation; text in one language is fed to SMT systems and the other is used as human-generated sentences.

the effect of the sizes of  $N$ -grams and development dataset in the experiments.

Using the proposed features, we train an SVM classifier for detecting machine-translated sentences. We use an implementation of LIBSVM (Chang and Lin, 2011) with a radial basis function kernel due to the relatively small number of features in the proposed method. We set appropriate parameters by grid search in a preliminary experiment.

We evaluate the performance of MT detection based on accuracy<sup>6</sup> that is a broadly used evaluation metric for classification problems:

$$\text{accuracy} = \frac{n_{TP} + n_{TN}}{n},$$

where  $n_{TP}$  and  $n_{TN}$  are the numbers of true-positives and true-negatives, respectively, and  $n$  is the total number of exemplars. The accuracy scores that we report in Sec. 5 are all based on 10-fold cross validation.

### 4.3 Comparison Method

We compare our method with the method of (Moore and Lewis, 2010) (*Cross-Entropy*). Although the Cross-Entropy method is designed for the task of domain adaptation of an LM, our problem is a variant of their original problem and thus their method is directly relevant. In our context, the method computes the cross-entropy scores  $I_{MT}(s)$  and  $I_H(s)$  of an input sentence  $s$  against LMs trained on machine-translated and human-generated sentences. Cross-entropy and perplexity are monotonically related, as perplexity of  $s$  according to an LM  $M$  is simply ob-

<sup>6</sup>Although we also examine precision and recall of classification results, they are similar to accuracy reported in this paper.

Method		Accuracy
Cross-Entropy		90.7
Lexical Feature		87.8
Proposed feature	<b>Word LMs</b>	<b>94.1</b>
	POS LMs	91.3
	FW LMs	82.7
	GPs	85.7

Table 4: Accuracy (%) of individual features and comparison methods

tained by  $b^{I_M(s)}$  where  $I_M(s)$  is cross-entropy score and  $b$  is a base with regard to which the cross-entropy is measured. The method scores the sentence according to the cross-entropy difference, *i.e.*,  $I_{MT}(s) - I_H(s)$ , and decides that the sentence is machine-translated when the score is lower than a predefined threshold. The classification is performed by 10-fold cross validation. We find the best performing threshold on a training set and evaluate the accuracy with a test set using the determined threshold.

Additionally, we compare our method to a method that uses a feature indicating presence or absence of unigrams, which we call *Lexical Feature*. This feature is commonly used for translationese detection and shows the best performance as a single feature in (Baroni and Bernardini, 2005). It is also used by Rarrick et al. (2011) and shows the best performance by itself in detecting machine-translated sentences in English-Japanese translation in the setting of bilingual input. We implement the feature and use it against a monolingual input to fit our problem setting.

## 5 Results and Discussions

In this section, we analyze and discuss the experiment results in detail.

### 5.1 Accuracy on Japanese Dataset

We evaluate the sentence-level and document-level accuracy of our method using the Japanese dataset. Specifically, we evaluate effects of individual features and their combinations, compare with human annotations, and assess performance variations across different sentence lengths and various settings on LM training.

**Effect of Individual Feature** Table 4 shows the accuracy scores of individual features and comparison methods. We refer to features for fluency ( $f_{w,H}$ ,  $f_{w,MT}$ ) as *Word LMs*, grammaticality using POS LMs ( $f_{pos,H}$ ,  $f_{pos,MT}$ ) as *POS LMs*

Method	Accuracy
Word LMs + GPs	94.7
Word LMs + POS LMs	95.1
Word LMs + POS LMs + GPs	95.4
Word LMs + POS LMs + FW LMs	95.5
<b>All</b>	<b>95.8</b>

Table 5: Accuracy (%) of feature combinations; there are significant differences ( $p \ll .01$ ) against the accuracy score of Word LMs.

and function word LMs ( $f_{fw,H}$ ,  $f_{fw,MT}$ ) as *FW LMs*, respectively, and for completeness of gappy-phrases ( $f_{g,H}$ ,  $f_{g,MT}$ ) as *GPs*. The Word LMs show the best accuracy that outperforms Cross-Entropy by 3.4% and Lexical Feature by 6.3%. This high accuracy is achieved by contrasting fluency in human-generated and machine-translated text to capture the phrase salad phenomenon. The accuracy of Word LM trained only on human-generated sentences is limited to 65.5%. On the other hand, the accuracy of Word LM trained on machine-translated sentences shows a better performance (84.4%). By combining these into a single feature vector  $f = (f_{w,H}, f_{w,MT}, f_{len})$ , the accuracy is largely improved.

It is interesting that Lexical Feature achieves a high accuracy of 87.8% despite its simplicity. Since Lexical Feature is a bag-of-words model, it can consider distant words in a sentence. This is effective for capturing a phrase salad that occurs among distant phrases, which  $N$ -gram cannot cover. As for Cross-Entropy, a simple subtraction of cross-entropy scores cannot well contrast the fluency in human-generated and machine-translated text and results in poorer accuracy than Word LMs.

The accuracy of POS LMs (91.3%) is slightly lower than that of Word LMs due to the limited vocabulary, *i.e.*, the number of POSs. The accuracy of FW LMs and GPs are even lower. This is convincing since these features cannot have reasonable values when a sentence does not include a function word and gappy-phrase. However, these features are complementary to Word LMs as we will see in the next paragraph.

**Effect of Feature Combination** Table 5 shows the accuracy when combining features. Sign tests show that the accuracy scores of these feature combinations are significantly different ( $p \ll .01$ ) against the accuracy of Word LMs. The results show that the features complement each other. The

Error	Ratio (%)	Accuracy	
		Word LMs	All
Has wrong content words	37.8	93.1	95.0
Misses content words	12.2	91.8	96.5
Has wrong function words	19.7	92.7	97.1
Misses function words	13.0	93.3	95.6
Has wrong inflections	10.8	97.3	98.7

Table 6: Distribution (%) of machine translation errors and accuracy (%) of proposed method on the different errors

combination of all features reaches an accuracy of 95.8%, which improves the accuracy of Word LMs by 1.7%. This result supports that FW LMs and GPs are effective to capture a phrase salad occurring in distant phrases and complement the evidence in  $N$ -grams that is captured by LMs. This effect becomes more obvious in the human evaluation.

We also evaluate the accuracy of the proposed method at a document level. Due to the high accuracy at the sentence-level, we use a voting method to judge a document, *i.e.*, deciding if the document is machine-translated when  $\gamma\%$  of its sentences are judged as machine-translated. We use all features and find that our method achieves 99% precision and recall with  $\gamma = 50$ .

**Human Evaluation** To further investigate the characteristics of our method, we conduct a human evaluation. We sample Japanese sentences and ask three native speakers to 1) judge whether a sentence is human-generated or machine-translated and 2) list errors that the sentence contains. Regarding the task 1), we allow the annotators to assign “hard to determine” for difficult cases. We allocate about 230 sentences for each annotator (in total 700 sentences) without overlapping annotation sets.

The accuracy of annotations is found to be 88.2%, which shows that our method is even superior to native speakers. Agreement between the annotators and our method (with all features) is 85.1%. As we interview the annotators, we find that human annotations are strongly affected by the annotators’ domain knowledge. For example, technical sentences are more often misclassified by the annotators.

Table 6 shows the distribution of errors on machine-translated sentences found by the annotators (on sentences that they correctly classified) with the accuracy of Word LMs and all features on

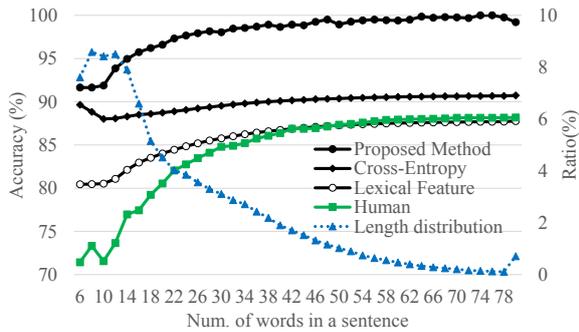


Figure 3: Accuracy (%) across different sentence lengths (the primary axis) and distribution (%) of sentence lengths in the evaluation dataset (the secondly axis)

these sentences (a sentence may contain multiple errors). It indicates that the accuracy of Word LMs is improved by feature combination; from 1.4% on sentences of “Has wrong inflections” to 4.7% on sentences of “Misses content words”.

**Effect of Sentence Length** The accuracy of the proposed method is significantly affected by sentence length (the number of words in a sentence). Fig. 3 shows the accuracy of the proposed method (with all features) and comparison methods w.r.t. sentence lengths (with the primary axis), as well as the distribution of sentence lengths in the evaluation dataset (with the secondly axis). We aggregate the classification results on each cross-validation (test results). It also shows the accuracy of human annotations w.r.t. sentence lengths, which we obtain for the 700 sentences in the human evaluation. The accuracy drops on all methods when sentences are short; the accuracy of our method is 91.6% when a sentence contains less than or equal to 10 words. The proposed method shows the similar trend with the human annotations, and even the accuracy of human annotations significantly drops on such short sentences. This result indicates that SMT results on short sentences tend to be of sufficient quality and indistinguishable from human-generated sentences. Since such high-quality machine-translations do not harm the quality of Web-mined data, we do not need to detect them.

**Effect of Setting on LM Training** We evaluate the performance variation w.r.t. the sizes of  $N$ -grams and development dataset. Fig. 4 shows the accuracy of the LM based features and feature combination when changing sizes of  $N$ -grams. The performance of Word LMs is stabilized after

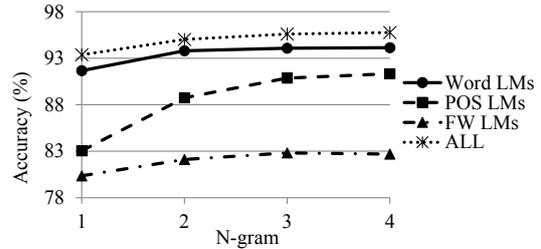


Figure 4: Effect of the sizes of  $N$ -grams on MT detection accuracy (%)

3-gram while that of POS LMs is still improved at 4-gram. This is because POS LMs need more evidence to compensate for their limited vocabulary. FW LMs become stable at 3-gram because the possible number of function words in a sentence should be small.

When we change the size of the development dataset with 10% increments, the accuracy curve is stabilized when the size is 40% of all set. Considering the fact that the overall development dataset is small, it shows that our method is deployable with a small dataset.

## 5.2 Accuracy on English Dataset

To investigate the applicability of our method to other languages, we apply the same method to the English dataset. Because English is a configurational language, function words are less flexible than case markers in Japanese. Therefore, SMT systems may better handle English function words, which potentially decreases the effect of FW LMs in our method. In addition, because English is a morphologically poor language, the effect of POS LMs may be reduced.

Nevertheless, in our experiment, all features are shown to be effective even with the English dataset. The combination of all features achieves the best performance, with an accuracy of 93.1%, which outperforms Cross-Entropy by 1.9%, and Lexical Feature by 8.5%. Even though improvements by POS LMs and FW LMs are smaller than Japanese case, their effects are still positive. We also find that GPs stably contribute to the accuracy. These results show the applicability of our method to other languages.

## 5.3 Accuracy on Raw Web Pages

To avoid unmodeled factors affecting the evaluation, we have carefully removed noise from our experiment datasets. However, real Web pages are

more complex; there are often instances of sentence fragments, such as captions and navigational link text. To evaluate the accuracy of our method on real Web pages, we conduct experiments using the dataset generated by Rarrick et al. (2011) that contains randomly crawled Web pages annotated by two annotators to judge if a page is human-generated or machine-translated. We use Japanese sentences extracted from 69 pages (43 human-generated and 26 machine-translated pages) where the annotators' judgments agree; 3,312 sentences consisting of 1,399 machine-translated and 1,913 human-generated sentences. To replicate the situation in real Web pages, we conduct a minimal preprocessing, *i.e.*, simply removing HTML tags, and then feed all the remaining text to our method.

An SVM classifier is trained with features obtained by the LMs and gappy-phrases computed from the data described in Sec. 4.1. Our method shows 80.6% accuracy at a sentence level and 82.4% accuracy at a document level using the voting method. One factor for this performance difference is again sentence lengths, as SMT results of short phrases in Web pages can be of high-quality. Another factor is the noise in Web pages. We find that experimental pages contain lots of non-sentences, such as fragments of scripts and product codes. The results show that we need a preprocessing to remove typical noise in Web text before SMT detection to handle noisy Web pages.

#### 5.4 Quality of Cleaned Data

Finally, we briefly demonstrate the effect of machine-translation filtering in an end-to-end scenario, taking LM construction as an example. We construct LMs reusing the Japanese evaluation dataset described in Sec. 4.1 where machine-translated sentences are removed by the proposed method (LM-Proposed), Lexical Feature (LM-LF), and Cross-Entropy (LM-CE), as well as an LM with all sentences, *i.e.*, with machine-translated sentences (LM-All). As a result of 5-fold cross-validation, LM-Proposed has 17.8%, 17.1%, and 16.3% lower perplexities on average compared to LM-All, LM-LF, and LM-CE, respectively. These results show that our method is useful for improving the quality of Web-mined data.

## 6 Conclusion

We propose a method for detecting machine-translated sentences from monolingual Web-text focusing on the phrase salad phenomenon produced by existing SMT systems. The experimental results show that our method achieves an accuracy of 95.8% for sentences and 80.6% for noisy Web text.

We plan to extend our method to detect machine-translated sentences produced by different MT systems, *e.g.*, a rule-based system, and develop a unified framework for cleaning various types of noise in Web-mined data. In addition, we will investigate the effect of source and target languages on translation in terms of MT detection. As Lopez (2008) describes, a phrase-salad is a common phenomenon that characterizes current SMT results. Therefore, we expect that our method is basically effective on different language pairs. We will conduct experiments to evaluate performance difference using various language pairs.

## Acknowledgments

We sincerely appreciate Spencer Rarrick and Will Lewis for active discussion and sharing the experimental data with us. We thank Junichi Tsujii for his valuable feedback to improve our work.

## References

- Alexandra Antonova and Alexey Misyurev. 2011. Building a web-based parallel corpus and filtering out machine translated text. In *Proceedings of the Workshop on Building and Using Comparable Corpora*, pages 136–144.
- Eleftherios Avramidis, Maja Popovic, David Vilar Torres, and Aljoscha Burchardt. 2011. Evaluate with confidence estimation: Machine ranking of translation outputs using grammatical features. In *Proceedings of the Workshop on Statistical Machine Translation (WMT 2011)*, pages 65–70.
- Mohit Bansal, Chris Quirk, and Robert C. Moore. 2011. Gappy phrasal alignment by agreement. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 1308–1317.
- Marco Baroni and Silvia Bernardini. 2005. A new approach to the study of translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274.

- Chih-Chung Chang and Chih-Jen Lin. 2011. LIB-SVM : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 263–270.
- Simon Corston-Oliver, Michael Gamon, and Chris Brockett. 2001. A machine learning approach to the automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2001)*, pages 148–155.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1535–1545.
- Michael Gamon, Anthony Aue, and Martine Smets. 2005. Sentence-level MT evaluation without reference translations: Beyond language modeling. In *Proceedings of European Association for Machine Translation (EAMT 2005)*.
- Google N-gram Corpus. 2006. <http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13>.
- Google Translate. 2006. <http://code.google.com/apis/language/>.
- Iustina Ilisei, Diana Inkpen, Gloria Corpas Pastor, and Ruslan Mitkov. 2010. Identification of translationese: A machine learning approach. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics (CI-Ling 2010)*, pages 503–511.
- Tatsuya Ishisaka, Masao Utiyama, Eiichiro Sumita, and Kazuhide Yamamoto. 2009. Development of a Japanese-English software manual parallel corpus. In *Proceedings of the Machine Translation Summit (MT Summit XII)*.
- Long Jiang, Shiquan Yang, Ming Zhou, Xiaohua Liu, and Qingsheng Zhu. 2009. Mining bilingual data from the web with adaptively learnt patterns. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, pages 870–878.
- Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pages 230–237.
- David Kurokawa, Cyril Goutte, and Pierre Isabelle. 2009. Automatic detection of translated text and its impact on machine translation. In *Proceedings of the Machine Translation Summit (MT-Summit XII)*.
- Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers.
- Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys*, 40(3):1–49.
- Xiaoyi Ma. 2006. Champollion: a robust parallel text sentence aligner. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2006)*, pages 489–492.
- Microsoft Translator. 2009. <http://www.microsofttranslator.com/dev/>.
- Microsoft Web N-gram Services. 2010. <http://research.microsoft.com/web-ngram>.
- Robert Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 220–224.
- Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A taxonomy of relational patterns with semantic types. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1135–1145.
- Jian-Yun Nie, Michel Simard, Pierre Isabelle, and Richard Durand. 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *Proceedings of the Annual International ACM SIGIR Conference (SIGIR 1999)*, pages 74–81.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 311–318.
- Kristen Parton, Joel Tetreault, Nitin Madnani, and Martin Chodorow. 2011. E-rating machine translation. In *Proceedings of the Workshop on Statistical Machine Translation (WMT 2011)*, pages 108–115.
- Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. 2001. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the International Conference on Data Engineering (ICDE 2001)*, pages 215–224.

- Spencer Rarrick, Chris Quirk, and Will Lewis. 2011. MT detection in web-scraped parallel corpora. In *Proceedings of the Machine Translation Summit (MT Summit XIII)*.
- Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47.
- Lei Shi, Cheng Niu, Ming Zhou, and Jianfeng Gao. 2006. A DOM tree alignment model for mining parallel data from the web. In *Proceedings of the International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 489–496.
- Andreas Stolcke. 2002. SRILM—an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP 2002)*, pages 901–904.
- Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of International Conference on World Wide Web (WWW 2007)*, pages 697–706.
- Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003)*, pages 252–259.
- Vladimir N. Vapnik. 1995. The nature of statistical learning theory. *Springer*.
- Jun Zhu, Zaiqing Nie, Xiaojiang Liu, Bo Zhang, and Ji-Rong Wen. 2009. StatSnowball: a statistical approach to extracting entity relationships. In *Proceedings of International Conference on World Wide Web (WWW 2009)*, pages 101–110.

# Paraphrase-Driven Learning for Open Question Answering

Anthony Fader   Luke Zettlemoyer   Oren Etzioni

Computer Science & Engineering

University of Washington

Seattle, WA 98195

{afader, lsz, etzioni}@cs.washington.edu

## Abstract

We study question answering as a machine learning problem, and induce a function that maps open-domain questions to queries over a database of web extractions. Given a large, community-authored, question-paraphrase corpus, we demonstrate that it is possible to learn a semantic lexicon and linear ranking function without manually annotating questions. Our approach automatically generalizes a seed lexicon and includes a scalable, parallelized perceptron parameter estimation scheme. Experiments show that our approach more than quadruples the recall of the seed lexicon, with only an 8% loss in precision.

## 1 Introduction

Open-domain question answering (QA) is a long-standing, unsolved problem. The central challenge is to automate every step of QA system construction, including gathering large databases and answering questions against these databases. While there has been significant work on large-scale information extraction (IE) from unstructured text (Banko et al., 2007; Hoffmann et al., 2010; Riedel et al., 2010), the problem of answering questions with the noisy knowledge bases that IE systems produce has received less attention. In this paper, we present an approach for learning to map questions to formal queries over a large, open-domain database of extracted facts (Fader et al., 2011).

Our system learns from a large, noisy, question-paraphrase corpus, where question clusters have a common but unknown query, and can span a diverse set of topics. Table 1 shows example paraphrase clusters for a set of factual questions. Such data provides strong signal for learning about lexical variation, but there are a number

Who wrote the Winnie the Pooh books? Who is the author of winnie the pooh? What was the name of the authur of winnie the pooh? Who wrote the series of books for Winnie the poo? Who wrote the children’s storybook ‘Winnie the Pooh’? Who is poohs creator?
What relieves a hangover? What is the best cure for a hangover? The best way to recover from a hangover? Best remedy for a hangover? What takes away a hangover? How do you lose a hangover? What helps hangover symptoms?
What are social networking sites used for? Why do people use social networking sites worldwide? Advantages of using social network sites? Why do people use social networks a lot? Why do people communicate on social networking sites? What are the pros and cons of social networking sites?
How do you say Santa Claus in Sweden? Say santa clause in sweden? How do you say santa clause in swedish? How do they say santa in Sweden? In Sweden what is santa called? Who is sweden santa?

Table 1: Examples of paraphrase clusters from the WikiAnswers corpus. Within each cluster, there is a wide range of syntactic and lexical variations.

of challenges. Given that the data is community-authored, it will inevitably be incomplete, contain incorrectly tagged paraphrases, non-factual questions, and other sources of noise.

Our core contribution is a new learning approach that scalably sifts through this paraphrase noise, learning to answer a broad class of factual questions. We focus on answering open-domain questions that can be answered with single-relation queries, *e.g.* all of the paraphrases of “Who wrote Winnie the Pooh?” and “What cures a hangover?” in Table 1. The algorithm answers such questions by mapping them to executable queries over a tuple store containing relations such as `authored(milne, winnie-the-pooh)` and `treat(bloody-mary, hangover-symptoms)`.

The approach automatically induces lexical structures, which are combined to build queries for unseen questions. It learns lexical equivalences for relations (e.g., *wrote*, *authored*, and *creator*), entities (e.g., *Winnie the Pooh* or *Pooh Bear*), and question templates (e.g., *Who  $\tau$  the  $e$  books?* and *Who is the  $\tau$  of  $e$ ?*). Crucially, the approach does not require any explicit labeling of the questions in our paraphrase corpus. Instead, we use 16 seed question templates and string-matching to find high-quality queries for a small subset of the questions. The algorithm uses learned word alignments to aggressively generalize the seeds, producing a large set of possible lexical equivalences. We then learn a linear ranking model to filter the learned lexical equivalences, keeping only those that are likely to answer questions well in practice.

Experimental results on 18 million paraphrase pairs gathered from WikiAnswers<sup>1</sup> demonstrate the effectiveness of the overall approach. We performed an end-to-end evaluation against a database of 15 million facts automatically extracted from general web text (Fader et al., 2011). On known-answerable questions, the approach achieved 42% recall, with 77% precision, more than quadrupling the recall over a baseline system.

In sum, we make the following contributions:

- We introduce PARALEX, an end-to-end open-domain question answering system.
- We describe scalable learning algorithms that induce general question templates and lexical variants of entities and relations. These algorithms require no manual annotation and can be applied to large, noisy databases of relational triples.
- We evaluate PARALEX on the end-task of answering questions from WikiAnswers using a database of web extractions, and show that it outperforms baseline systems.
- We release our learned lexicon and question-paraphrase dataset to the research community, available at <http://openie.cs.washington.edu>.

## 2 Related Work

Our work builds upon two major threads of research in natural language processing: information extraction (IE), and natural language interfaces to databases (NLIDB).

<sup>1</sup><http://wiki.answers.com/>

Research in IE has been moving towards the goal of extracting facts from large text corpora, across many domains, with minimal supervision (Mintz et al., 2009; Hoffmann et al., 2010; Riedel et al., 2010; Hoffmann et al., 2011; Banko et al., 2007; Yao et al., 2012). While much progress has been made in converting text into structured knowledge, there has been little work on answering natural language questions over these databases. There has been some work on QA over web text (Kwok et al., 2001; Brill et al., 2002), but these systems do not operate over extracted relational data.

The NLIDB problem has been studied for decades (Grosz et al., 1987; Katz, 1997). More recently, researchers have created systems that use machine learning techniques to automatically construct question answering systems from data (Zelle and Mooney, 1996; Popescu et al., 2004; Zettlemoyer and Collins, 2005; Clarke et al., 2010; Liang et al., 2011). These systems have the ability to handle questions with complex semantics on small domain-specific databases like GeoQuery (Tang and Mooney, 2001) or subsets of Freebase (Cai and Yates, 2013), but have yet to scale to the task of general, open-domain question answering. In contrast, our system answers questions with more limited semantics, but does so at a very large scale in an open-domain manner. Some work has been made towards more general databases like DBpedia (Yahya et al., 2012; Unger et al., 2012), but these systems rely on hand-written templates for question interpretation.

The learning algorithms presented in this paper are similar to algorithms used for paraphrase extraction from sentence-aligned corpora (Barzilay and McKeown, 2001; Barzilay and Lee, 2003; Quirk et al., 2004; Bannard and Callison-Burch, 2005; Callison-Burch, 2008; Marton et al., 2009). However, we use a paraphrase corpus for extracting lexical items relating natural language patterns to database concepts, as opposed to relationships between pairs of natural language utterances.

## 3 Overview of the Approach

In this section, we give a high-level overview of the rest of the paper.

**Problem** Our goal is to learn a function that will map a natural language question  $x$  to a query  $z$  over a database  $D$ . The database  $D$  is a collection of assertions in the form  $r(e_1, e_2)$  where  $r$  is a bi-

nary relation from a vocabulary  $R$ , and  $e_1$  and  $e_2$  are entities from a vocabulary  $E$ . We assume that the elements of  $R$  and  $E$  are human-interpretable strings like `population` or `new-york`. In our experiments,  $R$  and  $E$  contain millions of entries representing ambiguous and overlapping concepts. The database is equipped with a simple interface that accepts queries in the form  $r(?, e_2)$  or  $r(e_1, ?)$ . When executed, these queries return all entities  $e$  that satisfy the given relationship. Thus, our task is to find the query  $z$  that best captures the semantics of the question  $x$ .

**Model** The question answering model includes a lexicon and a linear ranking function. The lexicon  $L$  associates natural language patterns to database concepts, thereby defining the space of queries that can be derived from the input question (see Table 2). Lexical entries can pair strings with database entities (`nyc` and `new-york`), strings with database relations (`big` and `population`), or question patterns with templated database queries (`how r is e?` and `r(?, e)`). We describe this model in more detail in Section 4.

**Learning** The learning algorithm induces a lexicon  $L$  and estimates the parameters  $\theta$  of the linear ranking function. We learn  $L$  by bootstrapping from an initial seed lexicon  $L_0$  over a corpus of question paraphrases  $\mathcal{C} = \{(x, x') : x' \text{ is a paraphrase of } x\}$ , like the examples in Table 1. We estimate  $\theta$  by using the initial lexicon to automatically label queries in the paraphrase corpus, as described in Section 5.2. The final result is a scalable learning algorithm that requires no manual annotation of questions.

**Evaluation** In Section 8, we evaluate our system against various baselines on the end-task of question answering against a large database of facts extracted from the web. We use held-out known-answerable questions from WikiAnswers as a test set.

## 4 Question Answering Model

To answer questions, we must find the best query for a given natural language question.

### 4.1 Lexicon and Derivations

To define the space of possible queries, PARALEX uses a lexicon  $L$  that encodes mappings from natural language to database concepts (entities, relations, and queries). Each entry in  $L$  is a pair  $(p, d)$

Entry Type	NL Pattern	DB Concept
Entity	<code>nyc</code>	<code>new-york</code>
Relation	<code>big</code>	<code>population</code>
Question (1-Arg.)	<code>how big is e</code>	<code>population(?, e)</code>
Question (2-Arg.)	<code>how r is e</code>	<code>r(?, e)</code>

Table 2: Example lexical entries.

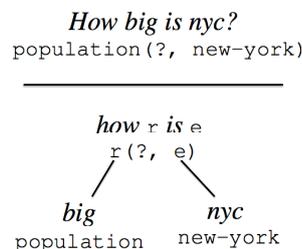
where  $p$  is a pattern and  $d$  is an associated database concept. Table 2 gives examples of the entry types in  $L$ : entity, relation, and question patterns.

**Entity patterns** match a contiguous string of words and are associated with some database entity  $e \in E$ .

**Relation patterns** match a contiguous string of words and are associated with a relation  $r \in R$  and an argument ordering (e.g. the string `child` could be modeled as either `parent-of` or `child-of` with opposite argument ordering).

**Question patterns** match an entire question string, with gaps that recursively match an entity or relation patterns. Question patterns are associated with a templated database query, where the values of the variables are determined by the matched entity and relation patterns. A question pattern may be **1-Argument**, with a variable for an entity pattern, or **2-Argument**, with variables for an entity pattern and a relation pattern. A 2-argument question pattern may also invert the argument order of the matched relation pattern, e.g. `who r e?` may have the opposite argument order of `who did e r?`

The lexicon is used to generate a derivation  $y$  from an input question  $x$  to a database query  $z$ . For example, the entries in Table 2 can be used to make the following derivation from the question `How big is nyc?` to the query `population(?, new-york)`:



This derivation proceeds in two steps: first matching a question form like `How r is e?` and then mapping `big` to `population` and `nyc` to `new-york`. Factoring the derivation this way allows the lexical entries for `big` and `nyc` to be reused in semanti-

cally equivalent variants like *nyc how big is it?* or *approximately how big is nyc?* This factorization helps the system generalize to novel questions that do not appear in the training set.

We model a derivation as a set of  $(p_i, d_i)$  pairs, where each  $p_i$  matches a substring of  $x$ , the substrings cover all words in  $x$ , and the database concepts  $d_i$  compose to form  $z$ . Derivations are rooted at either a 1-argument or 2-argument question entry and have entity or relation entries as leaves.

## 4.2 Linear Ranking Function

In general, multiple queries may be derived from a single input question  $x$  using a lexicon  $L$ . Many of these derivations may be incorrect due to noise in  $L$ . Given a question  $x$ , we consider all derivations  $y$  and score them with  $\theta \cdot \phi(x, y)$ , where  $\phi(x, y)$  is a  $n$ -dimensional feature representation and  $\theta$  is a  $n$ -dimensional parameter vector. Let  $\text{GEN}(x; L)$  be the set of all derivations  $y$  that can be generated from  $x$  using  $L$ . The best derivation  $y^*(x)$  according to the model  $(\theta, L)$  is given by:

$$y^*(x) = \arg \max_{y \in \text{GEN}(x; L)} \theta \cdot \phi(x, y)$$

The best query  $z^*(x)$  can be computed directly from the derivation  $y^*(x)$ .

Computing the set  $\text{GEN}(x; L)$  involves finding all 1-Argument and 2-Argument question patterns that match  $x$ , and then enumerating all possible database concepts that match entity and relation strings. When the database and lexicon are large, this becomes intractable. We prune  $\text{GEN}(x; L)$  using the model parameters  $\theta$  by only considering the  $N$ -best question patterns that match  $x$ , before additionally enumerating any relations or entities.

For the end-to-end QA task, we return a ranked list of answers from the  $k$  highest scoring queries. We score an answer  $a$  with the highest score of all derivations that generate a query with answer  $a$ .

## 5 Learning

PARALEX uses a two-part learning algorithm; it first induces an overly general lexicon (Section 5.1) and then learns to score derivations to increase accuracy (Section 5.2). Both algorithms rely on an initial seed lexicon, which we describe in Section 7.4.

### 5.1 Lexical Learning

The lexical learning algorithm constructs a lexicon  $L$  from a corpus of question paraphrases  $\mathcal{C} =$

$\{(x, x') : x' \text{ is a paraphrase of } x\}$ , where we assume that all paraphrased questions  $(x, x')$  can be answered with a single, initially unknown, query (Table 1 shows example paraphrases). This assumption allows the algorithm to generalize from the initial seed lexicon  $L_0$ , greatly increasing the lexical coverage.

As an example, consider the paraphrase pair  $x = \textit{What is the population of New York?}$  and  $x' = \textit{How big is NYC?}$  Suppose  $x$  can be mapped to a query under  $L_0$  using the following derivation  $y$ :

*what is the*  $r$  *of*  $e$  =  $r(? , e)$   
*population* =  $\text{population}$   
*new york* =  $\text{new-york}$

We can induce new lexical items by aligning the patterns used in  $y$  to substrings in  $x'$ . For example, suppose we know that the words in  $(x, x')$  align in the following way:

What is the population of New York?  
 \ / | / \ / \ /  
 How big is NYC?

Using this information, we can hypothesize that *how*  $r$  *is*  $e$ , *big*, and *nyc* should have the same interpretations as *what is the*  $r$  *of*  $e$ , *population*, and *new york*, respectively, and create the new entries:

*how*  $r$  *is*  $e$  =  $r(? , e)$   
*big* =  $\text{population}$   
*nyc* =  $\text{new-york}$

We call this procedure  $\text{InduceLex}(x, x', y, A)$ , which takes a paraphrase pair  $(x, x')$ , a derivation  $y$  of  $x$ , and a word alignment  $A$ , and returns a new set of lexical entries. Before formally describing  $\text{InduceLex}$  we need to introduce some definitions.

Let  $n$  and  $n'$  be the number of words in  $x$  and  $x'$ . Let  $[k]$  denote the set of integers  $\{1, \dots, k\}$ . A word alignment  $A$  between  $x$  and  $x'$  is a subset of  $[n] \times [n']$ . A phrase alignment is a pair of index sets  $(I, I')$  where  $I \subseteq [n]$  and  $I' \subseteq [n']$ . A phrase alignment  $(I, I')$  is consistent with a word alignment  $A$  if for all  $(i, i') \in A$ ,  $i \in I$  if and only if  $i' \in I'$ . In other words, a phrase alignment is consistent with a word alignment if the words in the phrases are aligned only with each other, and not with any outside words.

We will now define  $\text{InduceLex}(x, x', y, A)$  for the case where the derivation  $y$  consists of a 2-argument question entry  $(p_q, d_q)$ , a relation entry

**function** LEARNLEXICON**Inputs:**

- A corpus  $\mathcal{C}$  of paraphrases  $(x, x')$ . (Table 1)
- An initial lexicon  $L_0$  of (pattern, concept) pairs.
- A word alignment function  $\text{WordAlign}(x, x')$ . (Section 6)
- Initial parameters  $\theta_0$ .
- A function  $\text{GEN}(x; L)$  that derives queries from a question  $x$  using lexicon  $L$ . (Section 4)
- A function  $\text{InduceLex}(x, x', y, A)$  that induces new lexical items from the paraphrases  $(x, x')$  using their word alignment  $A$  and a derivation  $y$  of  $x$ . (Section 5.1)

**Output:** A learned lexicon  $L$ .

```

 $L = \{\}$ 
for all  $x, x' \in \mathcal{C}$  do
  if  $\text{GEN}(x; L_0)$  is not empty then
     $A \leftarrow \text{WordAlign}(x, x')$ 
     $y^* \leftarrow \arg \max_{y \in \text{GEN}(x; L_0)} \theta_0 \cdot \phi(x, y)$ 
     $L \leftarrow L \cup \text{InduceLex}(x, x', y^*, A)$ 
return  $L$ 

```

Figure 1: Our lexicon learning algorithm.

$(p_r, d_r)$ , and an entity entry  $(p_e, d_e)$ , as shown in the example above.<sup>2</sup>  $\text{InduceLex}$  returns the set of all triples  $(p'_q, d_q), (p'_r, d_r), (p'_e, d_e)$  such that for all  $p'_q, p'_r, p'_e$  such that

1.  $p'_q, p'_r, p'_e$  are a partition of the words in  $x'$ .
2. The phrase pairs  $(p_q, p'_q), (p_r, p'_r), (p_e, p'_e)$  are consistent with the word alignment  $A$ .
3. The  $p'_r$  and  $p'_e$  are contiguous spans of words in  $x'$ .

Figure 1 shows the complete lexical learning algorithm. In practice, for a given paraphrase pair  $(x, x')$  and alignment  $A$ ,  $\text{InduceLex}$  will generate multiple sets of new lexical entries, resulting in a lexicon with millions of entries. We use an existing statistical word alignment algorithm for  $\text{WordAlign}$  (see Section 6). In the next section, we will introduce a scalable approach for learning to score derivations to filter out lexical items that generalize poorly.

## 5.2 Parameter Learning

Parameter learning is necessary for filtering out derivations that use incorrect lexical entries like  $\text{new mexico} = \text{mexico}$ , which arise from noise in the paraphrases and noise in the word alignment.

<sup>2</sup> $\text{InduceLex}$  has similar behavior for the other type of derivation, which consists of a 1-argument question entry  $(p_q, d_q)$  and an entity  $(p_e, d_e)$ .

We use the hidden variable structured perceptron algorithm to learn  $\theta$  from a list of (question  $x$ , query  $z$ ) training examples. We adopt the iterative parameter mixing variation of the perceptron (McDonald et al., 2010) to scale to a large number of training examples.

Figure 2 shows the parameter learning algorithm. The parameter learning algorithm operates in two stages. First, we use the initial lexicon  $L_0$  to automatically generate (question  $x$ , query  $z$ ) training examples from the paraphrase corpus  $\mathcal{C}$ . Then we feed the training examples into the learning algorithm, which estimates parameters for the learned lexicon  $L$ .

Because the number of training examples is large, we adopt a parallel perceptron approach. We first randomly partition the training data  $\mathcal{T}$  into  $K$  equally-sized subsets  $\mathcal{T}_1, \dots, \mathcal{T}_K$ . We then perform perceptron learning on each partition in parallel. Finally, the learned weights from each parallel run are aggregated by taking a uniformly weighted average of each partition’s parameter vector. This procedure is repeated for  $T$  iterations.

The training data consists of (question  $x$ , query  $z$ ) pairs, but our scoring model is over (question  $x$ , derivation  $y$ ) pairs, which are unobserved in the training data. We use a hidden variable version of the perceptron algorithm (Collins, 2002), where the model parameters are updated using the highest scoring derivation  $y^*$  that will generate the correct query  $z$  using the learned lexicon  $L$ .

## 6 Data

For our database  $D$ , we use the publicly available set of 15 million REVERB extractions (Fader et al., 2011).<sup>3</sup> The database consists of a set of triples  $r(e_1, e_2)$  over a vocabulary of approximately 600K relations and 2M entities, extracted from the ClueWeb09 corpus.<sup>4</sup> The REVERB database contains a large cross-section of general world-knowledge, and thus is a good testbed for developing an open-domain QA system. However, the extractions are noisy, unnormalized (e.g., the strings `obama`, `barack-obama`, and `president-obama` all appear as distinct entities), and ambiguous (e.g., the relation `born-in` contains facts about both dates and locations).

<sup>3</sup>We used version 1.1, downloaded from <http://reverb.cs.washington.edu/>.

<sup>4</sup>The full set of REVERB extractions from ClueWeb09 contains over six billion triples. We used the smaller subset of triples to simplify our experiments.

```

function LEARNPARAMETERS
  Inputs:
  - A corpus  $\mathcal{C}$  of paraphrases  $(x, x')$ . (Table 1)
  - An initial lexicon  $L_0$  of (pattern, db concept) pairs.
  - A learned lexicon  $L$  of (pattern, db concept) pairs.
  - Initial parameters  $\theta_0$ .
  - Number of perceptron epochs  $T$ .
  - Number of training-data shards  $K$ .
  - A function  $\text{GEN}(x; L)$  that derives queries from a question  $x$  using lexicon  $L$ . (Section 4)
  - A function  $\text{PerceptronEpoch}(\mathcal{T}, \theta, L)$  that runs a single epoch of the hidden-variable structured perceptron algorithm on training set  $\mathcal{T}$  with initial parameters  $\theta$ , returning a new parameter vector  $\theta'$ . (Section 5.2)

  Output: A learned parameter vector  $\theta$ .

  // Step 1: Generate Training Examples  $\mathcal{T}$ 
   $\mathcal{T} = \{\}$ 
  for all  $x, x' \in \mathcal{C}$  do
    if  $\text{GEN}(x; L_0)$  is not empty then
       $y^* \leftarrow \arg \max_{y \in \text{GEN}(x; L_0)} \theta_0 \cdot \phi(x, y)$ 
       $z^* \leftarrow \text{query of } y^*$ 
      Add  $(x', z^*)$  to  $\mathcal{T}$ 

  // Step 2: Learn Parameters from  $\mathcal{T}$ 
  Randomly partition  $\mathcal{T}$  into shards  $\mathcal{T}_1, \dots, \mathcal{T}_K$ 
  for  $t = 1 \dots T$  do
    // Executed on  $k$  processors
     $\theta_{k,t} = \text{PerceptronEpoch}(\mathcal{T}_k, \theta_{t-1}, L)$ 
    // Average the weights
     $\theta_t = \frac{1}{K} \sum_k \theta_{k,t}$ 

  return  $\theta_T$ 

```

Figure 2: Our parameter learning algorithm.

Our paraphrase corpus  $\mathcal{C}$  was constructed from the collaboratively edited QA site WikiAnswers. WikiAnswers users can tag pairs of questions as alternate wordings of each other. We harvested a set of 18M of these question-paraphrase pairs, with 2.4M distinct questions in the corpus.

To estimate the precision of the paraphrase corpus, we randomly sampled a set of 100 pairs and manually tagged them as ‘paraphrase’ or ‘not-paraphrase.’ We found that 55% of the sampled pairs are valid paraphrased. Most of the incorrect paraphrases were questions that were related, but not paraphrased *e.g.* *How big is the biggest mall?* and *Most expensive mall in the world?*

We word-aligned each paraphrase pair using the MGIZA++ implementation of IBM Model 4 (Och and Ney, 2000; Gao and Vogel, 2008). The word-alignment algorithm was run in each direction  $(x, x')$  and  $(x', x)$  and then combined using the grow-diag-final-and heuristic (Koehn et al., 2003).

## 7 Experimental Setup

We compare the following systems:

- **PARALEX:** the full system, using the lexical learning and parameter learning algorithms from Section 5.
- **NoParam:** PARALEX without the learned parameters.
- **InitOnly:** PARALEX using only the initial seed lexicon.

We evaluate the systems’ performance on the end-task of QA on WikiAnswers questions.

### 7.1 Test Set

A major challenge for evaluation is that the REVERB database is incomplete. A system may correctly map a test question to a valid query, only to return 0 results when executed against the incomplete database. We factor out this source of error by semi-automatically constructing a sample of questions that are known to be answerable using the REVERB database, and thus allows for a meaningful comparison on the task of question understanding.

To create the evaluation set, we identified questions  $x$  in a held out portion of the WikiAnswers corpus such that (1)  $x$  can be mapped to some query  $z$  using an initial lexicon (described in Section 7.4), and (2) when  $z$  is executed against the database, it returns at least one answer. We then add  $x$  and all of its paraphrases as our evaluation set. For example, the question *What is the language of Hong-Kong* satisfies these requirements, so we added these questions to the evaluation set:

*What is the language of Hong-Kong?*  
*What language do people in hong kong use?*  
*How many languages are spoken in hong kong?*  
*How many languages hong kong people use?*  
*In Hong Kong what language is spoken?*  
*Language of Hong-kong?*

This methodology allows us to evaluate the systems’ ability to handle syntactic and lexical variations of questions that should have the same answers. We created 37 question clusters, resulting in a total of 698 questions. We removed all of these questions and their paraphrases from the training set. We also manually filtered out any incorrect paraphrases that appeared in the test clusters.

We then created a gold-standard set of  $(x, a, l)$  triples, where  $x$  is a question,  $a$  is an answer, and  $l$

Question Pattern	Database Query
<i>who</i> <sub>r</sub> <i>e</i>	$r(? , e)$
<i>what</i> <sub>r</sub> <i>e</i>	$r(? , e)$
<i>who</i> <i>does</i> <sub>e</sub> <i>r</i>	$r(e , ?)$
<i>what</i> <i>does</i> <sub>e</sub> <i>r</i>	$r(e , ?)$
<i>what is the</i> <sub>r</sub> <i>of</i> <i>e</i>	$r(? , e)$
<i>who is the</i> <sub>r</sub> <i>of</i> <i>e</i>	$r(? , e)$
<i>what is</i> <sub>r</sub> <i>by</i> <i>e</i>	$r(e , ?)$
<i>who is</i> <sub>e</sub> 's <sub>r</sub>	$r(? , e)$
<i>what is</i> <sub>e</sub> 's <sub>r</sub>	$r(? , e)$
<i>who is</i> <sub>r</sub> <i>by</i> <i>e</i>	$r(e , ?)$
<i>when did</i> <sub>e</sub> <i>r</i>	$r\text{-in}(e , ?)$
<i>when did</i> <sub>e</sub> <i>r</i>	$r\text{-on}(e , ?)$
<i>when was</i> <sub>e</sub> <i>r</i>	$r\text{-in}(e , ?)$
<i>when was</i> <sub>e</sub> <i>r</i>	$r\text{-on}(e , ?)$
<i>where was</i> <sub>e</sub> <i>r</i>	$r\text{-in}(e , ?)$
<i>where did</i> <sub>e</sub> <i>r</i>	$r\text{-in}(e , ?)$

Table 3: The question patterns used in the initial lexicon  $L_0$ .

is a label (correct or incorrect). To create the gold-standard, we first ran each system on the evaluation questions to generate  $(x, a)$  pairs. Then we manually tagged each pair with a label  $l$ . This resulted in a set of approximately 2,000 human judgments. If  $(x, a)$  was tagged with label  $l$  and  $x'$  is a paraphrase of  $x$ , we automatically added the labeling  $(x', a, l)$ , since questions in the same cluster should have the same answer sets. This process resulted in a gold standard set of approximately 48,000  $(x, a, l)$  triples.

## 7.2 Metrics

We use two types of metrics to score the systems. The first metric measures the precision and recall of each system’s highest ranked answer. Precision is the fraction of predicted answers that are correct and recall is the fraction of questions where a correct answer was predicted. The second metric measures the accuracy of the entire ranked answer set returned for a question. We compute the mean average precision (MAP) of each systems’ output, which measures the average precision over all levels of recall.

## 7.3 Features and Settings

The feature representation  $\phi(x, y)$  consists of indicator functions for each lexical entry  $(p, d) \in L$  used in the derivation  $y$ . For parameter learning, we use an initial weight vector  $\theta_0 = 0$ , use  $T = 20$

	F1	Precision	Recall	MAP
PARALEX	<b>0.54</b>	0.77	<b>0.42</b>	<b>0.22</b>
NoParam	0.30	0.53	0.20	0.08
InitOnly	0.18	<b>0.84</b>	0.10	0.04

Table 4: Performance on WikiAnswers questions known to be answerable using REVERB.

	F1	Precision	Recall	MAP
PARALEX	<b>0.54</b>	0.77	<b>0.42</b>	<b>0.22</b>
No 2-Arg.	0.40	<b>0.86</b>	0.26	0.12
No 1-Arg	0.35	0.81	0.22	0.11
No Relations	0.18	0.84	0.10	0.03
No Entity	0.36	0.55	0.27	0.15

Table 5: Ablation of the learned lexical items.

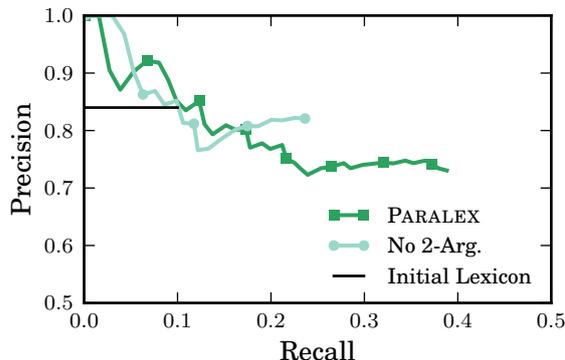


Figure 3: Precision-recall curves for PARALEX with and without 2-argument question patterns.

iterations and shard the training data into  $K = 10$  pieces. We limit each system to return the top 100 database queries for each test sentence. All input words are lowercased and lemmatized.

## 7.4 Initial Lexicon

Both the lexical learning and parameter learning algorithms rely on an initial seed lexicon  $L_0$ . The initial lexicon allows the learning algorithms to bootstrap from the paraphrase corpus.

We construct  $L_0$  from a set of 16 hand-written 2-argument question patterns and the output of the identity transformation on the entity and relation strings in the database. Table 3 shows the question patterns that were used in  $L_0$ .

## 8 Results

Table 4 shows the performance of PARALEX on the test questions. PARALEX outperforms the baseline systems in terms of both F1 and MAP. The lexicon-learning algorithm boosts the recall by a factor of 4 over the initial lexicon, showing the utility of the InduceLex algorithm. The

String	Learned Database Relations for String
<i>get rid of</i>	treatment-for, cause, get-rid-of, cure-for, easiest-way-to-get-rid-of
<i>word</i>	word-for, slang-term-for, definition-of, meaning-of, synonym-of
<i>speak</i>	speak-language-in, language-speak-in, principal-language-of, dialect-of
<i>useful</i>	main-use-of, purpose-of, importance-of, property-of, usefulness-of

String	Learned Database Entities for String
<i>smoking</i>	smoking, tobacco-smoking, cigarette, smoking-cigar, smoke, quit-smoking
<i>radiation</i>	radiation, electromagnetic-radiation, nuclear-radiation
<i>vancouver</i>	vancouver, vancouver-city, vancouver-island, vancouver-british-columbia
<i>protein</i>	protein, protein-synthesis, plasma-protein, monomer, dna

Table 6: Examples of relation and entity synonyms learned from the WikiAnswers paraphrase corpus.

parameter-learning algorithm also results in a large gain in both precision and recall: InduceLex generates a noisy set of patterns, so selecting the best query for a question is more challenging.

Table 5 shows an ablation of the different types of lexical items learned by PARALEX. For each row, we removed the learned lexical items from each of the types described in Section 4, keeping only the initial seed lexical items. The learned 2-argument question templates significantly increase the recall of the system. This increased recall came at a cost, lowering precision from 0.86 to 0.77. Thresholding the query score allows us to trade precision for recall, as shown in Figure 3. Table 6 shows some examples of the learned entity and relation synonyms.

The 2-argument question templates help PARALEX generalize over different variations of the same question, like the test questions shown in Table 7. For each question, PARALEX combines a 2-argument question template (shown below the questions) with the rules *celebrate* = holiday-of and *christians* = christians to derive a full query. Factoring the problem this way allows PARALEX to reuse the same rules in different syntactic configurations. Note that the imperfect training data can lead to overly-specific templates like *what are the religious r of e*, which can lower accuracy.

## 9 Error Analysis

To understand how close we are to the goal of open-domain QA, we ran PARALEX on an unrestricted sample of questions from WikiAnswers. We used the same methodology as described in the previous section, where PARALEX returns the top answer for each question using REVERB.

We found that PARALEX performs significantly worse on this dataset, with recall maxing out at ap-

Celebrations for Christians? <i>r for e?</i>
Celebrations of Christians? <i>r of e?</i>
What are some celebrations for Christians? <i>what are some r for e?</i>
What are some celebrations of the Christians? <i>what are some r of e?</i>
What are some of Christians celebrations? <i>what are some of e r?</i>
What celebrations do Christians do? <i>what r do e do?</i>
What did Christians celebrate? <i>what did e r?</i>
What are the religious celebrations of Christians? <i>what are the religious r of e?</i>
What celebration do Christians celebrate? <i>what r do e celebrate?</i>

Table 7: Questions from the test set with 2-argument question patterns that PARALEX used to derive a correct query.

proximately 6% of the questions answered at precision 0.4. This is not surprising, since the test questions are not restricted to topics covered by the REVERB database, and may be too complex to be answered by any database of relational triples.

We performed an error analysis on a sample of 100 questions that were either incorrectly answered or unanswered. We examined the candidate queries that PARALEX generated for each question and tagged each query as correct (would return a valid answer given a correct and complete database) or incorrect. Because the input questions are unrestricted, we also judged whether the questions could be faithfully represented as a  $r(?, e)$  or  $r(e, ?)$  query over the database vocabulary. Table 8 shows the distribution of errors.

The largest source of error (36%) were on com-

plex questions that could not be represented as a query for various reasons. We categorized these questions into groups. The largest group (14%) were questions that need n-ary or higher-order database relations, for example *How long does it take to drive from Sacramento to Cancun?* or *What do cats and dogs have in common?* Approximately 13% of the questions were how-to questions like *How do you make axes in minecraft?* whose answers are a sequence of steps, instead of a database entity. Lastly, 9% of the questions require database operators like joins, for example *When were Bobby Orr’s children born?*

The second largest source of error (32%) were questions that could be represented as a query, but where PARALEX was unable to derive any correct queries. For example, the question *Things grown on Nigerian farms?* was not mapped to any queries, even though the REVERB database contains the relation `grown-in` and the entity `nigeria`. We found that 13% of the incorrect questions were cases where the entity was not recognized, 12% were cases where the relation was not recognized, and 6% were cases where both the entity and relation were not recognized.

We found that 28% of the errors were cases where PARALEX derived a query that we judged to be correct, but returned no answers when executed against the database. For example, given the question *How much can a dietician earn?* PARALEX derived the query `salary-of(?, dietician)` but this returned no answers in the REVERB database.

Finally, approximately 4% of the questions included typos or were judged to be inscrutable, for example *Barovier hiriacy of evidence based for pressure sore?*

**Discussion** Our experiments show that the learning algorithms described in Section 5 allow PARALEX to generalize beyond an initial lexicon and answer questions with significantly higher accuracy. Our error analysis on an unrestricted set of WikiAnswers questions shows that PARALEX is still far from the goal of truly high-recall, open-domain QA. We found that many questions asked on WikiAnswers are either too complex to be mapped to a simple relational query, or are not covered by the REVERB database. Further, approximately one third of the missing recall is due to entity and relation recognition errors.

Incorrectly Answered/Unanswered Questions	
36%	Complex Questions <i>Need n-ary or higher-order relations (14%)</i> <i>Answer is a set of instructions (13%)</i> <i>Need database operators e.g. joins (9%)</i>
32%	Entity or Relation Recognition Errors <i>Entity recognition errors (13%)</i> <i>Relation recognition errors (12%)</i> <i>Entity &amp; relation recognition errors (7%)</i>
28%	Incomplete Database <i>Derived a correct query, but no answers</i>
4%	Typos/Inscrutable Questions

Table 8: Error distribution of PARALEX on an unrestricted sample of questions from the WikiAnswers dataset.

## 10 Conclusion

We introduced a new learning approach that induces a complete question-answering system from a large corpus of noisy question-paraphrases. Using only a seed lexicon, the approach automatically learns a lexicon and linear ranking function that demonstrated high accuracy on a held-out evaluation set.

A number of open challenges remain. First, precision could likely be improved by adding new features to the ranking function. Second, we would like to generalize the question understanding framework to produce more complex queries, constructed within a compositional semantic framework, but without sacrificing scalability. Third, we would also like to extend the system with other large databases like Freebase or DBpedia. Lastly, we believe that it would be possible to leverage the user-provided answers from WikiAnswers as a source of supervision.

## Acknowledgments

This research was supported in part by ONR grant N00014-11-1-0294, DARPA contract FA8750-09-C-0179, a gift from Google, a gift from Vulcan Inc., and carried out at the University of Washington’s Turing Center. We would like to thank Yoav Artzi, Tom Kwiatkowski, Yuval Marton, Mausam, Dan Weld, and the anonymous reviewers for their helpful comments.

## References

- Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open Information Extraction from the Web. In *Proceedings of the 20th international joint conference on Artificial intelligence*.
- Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*.
- Regina Barzilay and Lillian Lee. 2003. Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics*.
- Regina Barzilay and Kathleen R. McKeown. 2001. Extracting Paraphrases from a Parallel Corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*.
- Eric Brill, Susan Dumais, and Michele Banko. 2002. An Analysis of the AskMSR Question-Answering System. In *Proceedings of Empirical Methods in Natural Language Processing*.
- Qingqing Cai and Alexander Yates. 2013. Large-scale Semantic Parsing via Schema Matching and Lexicon Extension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- Chris Callison-Burch. 2008. Syntactic Constraints on Paraphrases Extracted from Parallel Corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*.
- James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving Semantic Parsing from the World's Response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*.
- Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying Relations for Open Information Extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Qin Gao and Stephan Vogel. 2008. Parallel Implementations of Word Alignment Tool. In *Proc. of the ACL 2008 Software Engineering, Testing, and Quality Assurance Workshop*.
- Barbara J. Grosz, Douglas E. Appelt, Paul A. Martin, and Fernando C. N. Pereira. 1987. TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces. *Artificial Intelligence*, 32(2):173–243.
- Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*.
- Boris Katz. 1997. Annotating the World Wide Web using Natural Language. In *RIAO*, pages 136–159.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics*.
- Cody Kwok, Oren Etzioni, and Daniel S. Weld. 2001. Scaling Question Answering to the Web. *ACM Trans. Inf. Syst.*, 19(3):242–262.
- Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning Dependency-Based Compositional Semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*.
- Yuval Marton, Chris Callison-Burch, and Philip Resnik. 2009. Improved Statistical Machine Translation Using Monolingually-Derived Paraphrases. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*.
- Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.
- Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant Supervision for Relation Extraction Without Labeled Data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL*.
- Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*.
- Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates. 2004. Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability. In *Proceedings of the Twentieth International Conference on Computational Linguistics*.
- Chris Quirk, Chris Brockett, and William Dolan. 2004. Monolingual Machine Translation for Paraphrase Generation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling Relations and Their Mentions without Labeled Text. In *Proceedings of the 2010 European conference on Machine learning and Knowledge Discovery in Databases*.
- Lappoon R. Tang and Raymond J. Mooney. 2001. Using Multiple Clause Constructors in Inductive Logic Programming for Semantic Parsing.
- Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-Based Question Answering over RDF Data. In *Proceedings of the 21st World Wide Web Conference 2012*.
- Mohamed Yahya, Klaus Berberich, Shady Elbasuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural Language Questions for the Web of Data. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.
- Limin Yao, Sebastian Riedel, and Andrew McCallum. 2012. Unsupervised Relation Discovery with Sense Disambiguation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*.
- John M. Zelle and Raymond J. Mooney. 1996. Learning to Parse Database Queries Using Inductive Logic Programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*.
- Luke S. Zettlemoyer and Michael Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorical Grammars. In *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence*.

# Aid is Out There: Looking for Help from Tweets during a Large Scale Disaster

István Varga<sup>†</sup> Motoki Sano<sup>†</sup> Kentaro Torisawa<sup>†</sup> Chikara Hashimoto<sup>†</sup>  
Kiyonori Ohtake<sup>†</sup> Takao Kawai<sup>§</sup> Jong-Hoon Oh<sup>†</sup> Stijn De Saeger<sup>†</sup>

<sup>†</sup>Information Analysis Laboratory,

National Institute of Information and Communications Technology (NICT), Japan  
{istvan, msano, torisawa, ch, kiyonori.ohtake, rovellia, stijn}@nict.go.jp

<sup>§</sup>Knowledge Discovery Research Laboratories, NEC Corporation, Japan

t-kawai@bx.jp.nec.com

## Abstract

The 2011 Great East Japan Earthquake caused a wide range of problems, and as countermeasures, many aid activities were carried out. Many of these problems and aid activities were reported via Twitter. However, most problem reports and corresponding aid messages were not successfully exchanged between victims and local governments or humanitarian organizations, overwhelmed by the vast amount of information. As a result, victims could not receive necessary aid and humanitarian organizations wasted resources on redundant efforts. In this paper, we propose a method for discovering matches between problem reports and aid messages. Our system contributes to problem-solving in a large scale disaster situation by facilitating communication between victims and humanitarian organizations.

## 1 Introduction

The 2011 Great East Japan Earthquake in March 11, 2011 killed 15,883 people and destroyed over 260,000 households (National Police Agency of Japan, 2013). Accustomed way of living suddenly became unmanageable and people found themselves in extreme conditions for months.

Just after the disaster, many people used Twitter for posting problem reports and aid messages as it functioned while most communication channels suffered disruptions (Winn, 2011; Acar and Muraki, 2011; Sano et al., 2012). Examples of such problem reports and aid messages, translated from Japanese tweets, are given below (P1, A1).

**P1** *My friend said infant formula is sold out. If somebody knows shops in Sendai-city where they still have it in stock, please let us know.*

**A1** *At Jusco supermarket in Sendai, you can still buy water and infant formula.*

If A1 would have been forwarded to the sender of P1, it could have helped since it would help the “friend” to obtain infant formula. But in reality, the majority of such reports/messages, especially unforeseen ones went unnoticed amongst the mass of information (Ohtake et al., 2013). In addition, there were cases where many humanitarian organizations responded to the same problems and wasted precious resources. For instance, many volunteers responded to problems which were heavily reported by public media, leading to oversupply (Saijo, 2012). Such waste of resources could have been avoided if the organizations would have successfully shared the aid messages for the same problems.

Such observations motivated this work. We developed methods for recognizing problem reports and aid messages in tweets and finding proper matches between them. By browsing the discovered matches, victims can be assisted to overcome their problems, and humanitarian organizations can avoid redundant relief efforts. We define problem reports, aid messages and their successful matches as follows.

**Problem report:** A tweet that informs about the possibility or emergence of a problem that requires a treatment or countermeasure.

**Aid message:** A tweet that (1) informs about situations or actions that can be a remedy or solution for a problem, or (2) informs that the problem is solved or is about to be solved.

**Problem-aid tweet match:** A tweet pair is a problem-aid tweet match (1) if the aid message informs how to overcome the problem, (2) if the aid message informs about the set-

tlement of the problem, or (3) if the aid message provides information which contributes to the settlement of the problem.

In this work we excluded *direct requests*, such as “Send us food!”, from problem reports. This is because it is relatively easy to recognize such direct requests by checking mood types (i.e., imperative) and their behavior is quite different from problem reports like “People in Sendai are starving”. Problem reports in this work do not directly state which actions are required, only implying the necessity of a countermeasure through claiming the existence of problems.

An underlying assumption of our method is that we can find a noun-predicate dependency relation that works as an *indicator* of problems and aids in problem reports and aid messages, which we refer to as *problem nucleus* and *aid nucleus*.<sup>1</sup> An example of problem nucleus is “infant formula is sold out” in P1, and that of aid nucleus is “(can) buy infant formula” in A1. Many problem-aid tweet matches can be recognized through problem and aid nuclei pairs.

We also assume that if the problem and aid nuclei match, they share the same noun. Then, the semantics of predicates in the nuclei is the main factor that decides whether the nuclei constitute a match. We introduce a semantic classification of predicates according to the framework of excitation polarities proposed in Hashimoto et al. (2012). Our hypothesis is that excitation polarities along with trouble expressions can characterize problem reports, aid messages and their matches. We developed a supervised method encoding such information into its features.

An evident alternative to this approach is to use sentiment analysis (Mandel et al., 2012; Tsagkaldou et al., 2011) assuming that problem reports should include something ‘bad’ while aid messages describe something ‘good’. However, we will show that this does not work well in our experiments. We think this is due to mismatch between the concepts of problem/aid and sentiment polarity. Note that previous work on ‘demand’ recognition also found similar tendencies (Kanayama and Nasukawa, 2008).

Another issue in this task is, of course, the context surrounding problem/aid nuclei. The fol-

<sup>1</sup>We found that out of 500 random tweets only 4.5% of problem reports and 9.1% of aid messages did not contain any problem report/aid message nuclei.

lowing (imaginary) tweets exemplify the problems caused by contexts.

**FP1** *I do not believe infant formula is sold out in Sendai.*

**FA1** *At Jusco supermarket in Iwaki, you can still buy infant formula.*

The problem nuclei of FP1 and P1 are the same but FP1 is not a problem report because of the expression “I do not believe”. The aid nuclei of FA1 and A1 are the same but FA1 does not constitute a proper match with P1 because FA1 and P1 refer to different cities, “Iwaki” and “Sendai”. In this work, the problems concerning the modality and other semantic modifications to problem/aid nuclei by context are dealt with by the introduction of features representing the text surrounding the nuclei in machine learning. As for the location problem, we apply a location recognizer to all tweets and restrict the matching candidates to the tweet pairs referring to the same location.

## 2 Approach

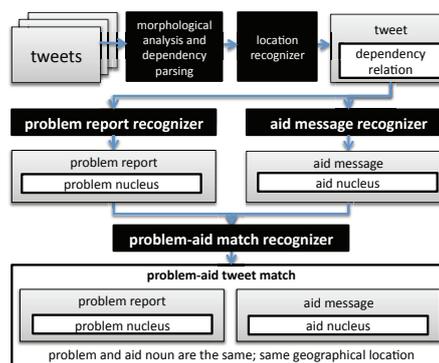


Figure 1: Problem-aid matching system overview.

We developed machine learning based systems to recognize problem reports, aid messages and problem-aid tweet matches. Figure 1 illustrates the whole system. First, location names in tweets are identified by matching tweets against our location dictionary, described in Section 3. Then, each tweet is paired with *each* dependency relation in the tweet, which is a candidate of problem/aid nuclei and given to the problem report and aid message recognizers. A tweet-nucleus-candidate pair judged as problem report is combined with another tweet-nucleus-candidate pair recognized as an aid message if the two nuclei share the same noun and the tweets share the same location name, and given to the problem-aid match recognizer.

In the following, problem and aid nuclei are denoted by a noun-template pair. A *template* is composed of a predicate and its argument position. For instance, “water supply stopped” in P2 is a problem nucleus, “water supply recovered” in A2 is an aid nucleus and they are denoted by the noun-template pairs ⟨water supply, X stopped⟩ and ⟨water supply, X recovered⟩.

**P2** *In Sendai city, water supply stopped.*

**A2** *In Sendai city, water supply recovered.*

Roughly speaking, we regard the tasks of problem report recognition and aid message recognition as the tasks of finding proper problem/aid nuclei in tweets and our method performs these tasks based on the semantic properties of nouns and templates in problem/aid nucleus candidates and their surrounding contexts.

The basic intuition behind this approach can be explained using excitation polarity proposed in Hashimoto et al. (2012). Excitation polarity differentiates *templates* into ‘excitatory’ or ‘inhibitory’ with regard to the main function or effect of entities referred to by their argument noun. While excitatory templates (e.g., cause X, buy X, suffer from X) entail that the main function or effect is activated or enhanced, inhibitory templates (e.g., ruin X, prevent X, X runs out) entail that the main function or effect is deactivated or suppressed. The templates that do not fit into the above categorization are classified as ‘neutral’.

We observed that problem reports in general included either of (A) a dependency relation between a noun referring to some trouble and an excitatory template or (B) a dependency relation between a noun not referring to any trouble and an inhibitory template. Examples of (A) include ⟨carbon monoxide poisoning, suffer from X⟩, ⟨false rumor, spread X⟩. They refer to events that activate troubles. On the other hand, (B) is exemplified by ⟨school, X is collapsed⟩, ⟨battery, X runs out⟩, which imply that some non-trouble objects such as resources, appliances and facilities are dysfunctional. We assume that if we can find such dependency relations in tweets, the tweets are likely to be problem reports.

Contrary, a tweet is more likely to be an aid message when it includes either (C) a dependency relation between a noun referring to some trouble and an inhibitory template or (D) a dependency relation between a noun not referring to any trou-

	trouble	non-trouble
<b>excitatory</b>	(A) problem nucleus	(D) aid nucleus
<b>inhibitory</b>	(C) aid nucleus	(B) problem nucleus

Table 1: Problem/aid-excitation matrix.

ble and an excitatory template. Examples of (C) are ⟨flu, X was eradicated (in some shelter)⟩ and ⟨debris, remove X⟩. They represent the *dysfunction* of troubles and can mean the solution or the settlement of troubles. On the other hand, examples of (D) include ⟨school, X re-build⟩ and ⟨baby formula, buy X⟩. They entail that some resources function properly or become available. These formulations are summarized in Table 1.

As an interesting consequence of such a view on problem/aid nucleus, we can say the following regarding problem-aid tweet matchings: when a problem nucleus and an aid nucleus are an adequate match, the excitation polarities of their templates are opposite. Consider the following tweets.

**P3** *Some people were going back to Iwaki, but the water system has not come back yet. It’s terrible that **bath is unusable**.*

**A3** *We **open the bath** for the public, located on the 2F of Iwaki Kuhon temple. If you’re staying at a relief shelter and would like to take a bath, you can use it.*

“Bath is unusable” in P3 is a problem nucleus while “open the bath” in A3 is an aid nucleus. Since the problem reported in P3 can be solved with A3, they are a successful match. The inhibitory template “X is unusable” indicates that the function of “bath”, a non-trouble expression, is suppressed. The excitatory template “open X” indicates that the function of “bath” is activated.

The same holds when we consider the noun referring to troubles like “flu”. The polarity of the template in a problem nucleus should be excitatory like “flu is raging” while that of an aid nucleus should be inhibitory like ⟨flu, X was eradicated⟩. These examples keep the constraint that the problem and aid nucleus should have opposite polarities when they constitute a match.

Note that the formulations of problem report, aid message and their matches or the excitation matrix (Table 1) were not presented to our annotators and our test/training data may contain data that contradict with the formulations. These formulations constitute the hypothesis to be validated in this work.

An important point to be stressed here is that there are problem-aid tweet matches that do not fit into our formulations. For instance, we assume that the problem nucleus and aid nucleus in a proper match share the same noun. However, tweet pairs such as “There are many injured people in Sendai city” and “We are sending ambulances to Sendai” can constitute a proper match, but there is no proper problem-aid nuclei pair that share the same noun in these tweets. (We can find the dependency relations sharing “Sendai” but they do not express anything about the contents of problem and aid.) The point is that the tweet pairs can be judged because people know ambulances can be a *countermeasure* to injured people as world knowledge. Introducing such world knowledge is beyond the scope of this current study.

Also, we exclude direct requests from problem reports. As mentioned in the introduction, identifying direct requests is relatively easy, hence we excluded them from our target.

### 3 Problem Report and Aid Message Recognizers

We recognize problem reports and aid messages in given tweets using a supervised classifier, SVMs with linear kernel, which worked best in our preliminary experiments. The feature set given to the SVMs are summarized in the top part of Table 2. Note that we used a common feature set for both the problem report recognizer and aid message recognizer and that it is categorized into several types: features concerning trouble expressions (TR), excitation polarity (EX), their combination (TRES1) and word sentiment polarity (WSP), features expressing morphological and syntactic structures of nuclei and their context surrounding problem/aid nuclei (MSA), features concerning semantic word classes (SWC) appearing in nuclei and their context, request phrases, such as “Please help us”, appearing in tweets (REQ), and geographical locations in tweets recognized by our location recognizer (GL). MSA is used to express the modality of nuclei and other contextual information surrounding nuclei. REQ was introduced based on our observation that if there are some requests in tweets, problem nuclei tend to appear as justification for the requests.

We also attempted to represent nucleus template IDs, noun IDs and their combinations directly in our feature set to capture typical templates fre-

TR	Whether the nucleus noun is a trouble/non-trouble expression.
EX1	The excitation polarity and the value of the excitation score of the nucleus template.
TRES1	All possible combinations of trouble/non-trouble of TR and excitation polarities of EX1.
WSP1	Whether the nucleus noun is positive/negative/not in the Word Sentiment Polarity (WSP) dictionary.
WSP2	Whether the nucleus template is positive/negative/not in the WSP dictionary.
WSP3	Whether the nucleus template is followed by a positive/negative word within the tweet.
MSA1	Morpheme $n$ -grams, syntactic dependency $n$ -grams in the tweet and morpheme $n$ -grams before and after the nucleus template. ( $1 \leq n \leq 3$ )
MSA2	Character $n$ -grams of the nucleus template to capture conjugation and modality variations. ( $1 \leq n \leq 3$ )
MSA3	Morpheme and part-of-speech $n$ -grams within the <i>bunsetsu</i> containing the nucleus template to capture conjugation and modality variations. ( $1 \leq n \leq 3$ ) (A <i>bunsetsu</i> is a syntactic constituent composed of a content word and several function words, the smallest unit of syntactic analysis in Japanese.)
MSA4	The part-of-speech of the nucleus template’s head to capture modality variations outside the nucleus template’s <i>bunsetsu</i> .
MSA5	The number of <i>bunsetsu</i> between the nucleus noun and the nucleus template. We found that a long distance between the noun and the template suggests parsing errors.
MSA6	Re-occurrence of the nucleus noun’s postpositional particle between the nucleus noun and the nucleus template. We found that the re-occurrence of the same postpositional particle within a clause suggests parsing errors.
SWC1	The semantic class $n$ -grams in the tweet.
SWC2	The semantic class(es) of the nucleus noun.
REQ	Presence of a request phrase in the tweet, identified from within 426 manually collected request phrases.
GL	Geographical locations in the tweet identified using our location recognizer. Existence/non-existence of locations in tweets are also encoded.
EX2	Whether the problem and aid nucleus templates have the same or opposite excitation polarities.
EX3	Product of the values of the excitation scores for the problem and the aid nucleus template.
TRES2	All possible combinations of trouble/non-trouble of TR, excitation polarity EX1 of the problem nucleus template and excitation polarity EX1 of the aid nucleus template.
SIM1	Common semantic word classes of the problem report and aid message.
SIM2	Whether there are common nouns modifying the common nucleus noun or not in the problem report and aid message.
SIM3	Whether the words in the same word class modify the common nucleus noun or not in the problem report and aid message.
SIM4	The semantic similarity score between the problem nucleus template and the aid nucleus template.
CTP	Whether the problem nucleus template and the aid nucleus template are in contradiction relation dictionary or not.
SSR1	Problem report recognizer’s SVM score of problem nucleus template.
SSR2	Problem report recognizer’s SVM score of aid nucleus template.
SSR3	Aid message recognizer’s SVM score of the problem nucleus template.
SSR4	Aid message recognizer’s SVM score of the aid nucleus template.

Table 2: Features used with the problem report recognizer and the aid message recognizer (*above*); additional features used in training the problem-aid match recognizer (*below*).

quently appearing in problem and aid nuclei, but since there was no improvement we omit them.

The other feature types need some non-trivial dictionaries. In the following, we explain how we created the dictionaries for each feature type along with the motivation behind their introduction.

**Trouble Expressions (TR)** As mentioned previously, trouble expressions work as good evidence for recognizing problem reports and aid messages. The TR feature indicates whether the noun in the problem/aid nucleus candidate is a trouble ex-

pression or not. For this purpose, we created a list of trouble expressions following the semi-supervised procedure presented in De Saeger et al. (2008). After manual validation of the list, we obtained 20,249 expressions referring to some troubles, such as “tsunami” and “flu”. The value of the TR feature is determined by checking whether the nucleus noun is contained in the list.

**Excitation Polarities (EX)** The excitation polarities are also important in recognizing problem reports and aid messages as mentioned before. For constructing the dictionary for excitation polarities of templates, we applied the bootstrapping procedure in Hashimoto et al. (2012) to 600 million Web pages. Hashimoto’s method provides the value of the *excitation score* in  $[-1, 1]$  for each template indicating the polarities and their *strength*. Positive value indicates excitatory, negative value inhibitory and small absolute value neutral. After manual checking of the results by the majority vote of three human annotators (other than the authors), we limited the templates to the ones that have score values consistent with the majority vote of the annotators, obtaining a dictionary consisting of 7,848 excitatory, 836 inhibitory and 7,230 neutral templates. The Fleiss’ (1971) kappa-score was 0.48 (moderate agreement). We used the excitation score values as feature values. Excitation has already been used in many works, such as causality and contradiction extraction (Hashimoto et al., 2012) or Why-QA (Oh et al., 2013).

**Word Sentiment Polarity (WSP)** As we suggested before, full-fledged sentiment analysis to recognize the expressions, including clauses and phrases, that refer to something good or bad was not effective in our task. However, the sentiment polarity, assigned to single words turned out to be effective. To identify the sentiment polarity of words, we employed the word sentiment polarity dictionary used with a sentiment analysis tool for Japanese, the Opinion Extraction Tool software<sup>2</sup>, which is an implementation of Nakagawa et al. (2010). The dictionary includes 9,030 positive and 27,951 negative words. Note that we used the Opinion Extraction Tool in the experiments to check the effectiveness of the full-fledged sentiment analysis in this task.

**Semantic Word Class (SWC)** We assume that nouns in the same semantic class behave simi-

<sup>2</sup>Provided at the ALAGIN Forum (<http://www.alagin.jp/>).

larly in crisis situations. For example, if “infection” appears in a problem report, the tweets including “pulmonary embolism” are also likely to be problem reports. Semantic word class features are used to capture such tendencies. We applied an EM-style word clustering algorithm in Kazama and Torisawa (2008) to 600 million Web pages and clustered 1 million nouns into 500 classes. This algorithm has been used in many works, such as relation extraction (De Saeger et al., 2011) and Why-QA (Oh et al., 2012), and can generate various kinds of semantically clean word classes, such as *foods*, *disease names*, and *natural disasters*. We used the word classes in tweets as features.<sup>3</sup>

**Geographical Locations (GL)** Our location recognizer matches tweets against our location dictionary. Location names and their existence/non-existence in tweets constitute evidence, thus we encoded such information into our features. The location dictionary was created from the Japan Post code data<sup>4</sup> and Wikipedia, containing 2.7 million location names including cities, schools and other facilities (Kazama et al., 2013).

## 4 Problem-Aid Match Recognizer

After problem report and aid message recognition, the positive outputs of the respective classifiers are used as input in this step. The problem-aid match recognizer classifies an aid message-nucleus pair together with the problem report-nucleus pair employing SVMs with linear kernel, which performed best in this task again. The problem-aid match recognizer uses all the features used in the problem report recognizer and the aid message recognizer along with additional features regarding: excitation polarity (EX) and trouble expressions (TR), distributional similarity (SIM), contradiction (CTP) and SVM-scores of the problem report and aid message recognizers (SSR). Here also we attempted to capture typical or frequent matches of nuclei using template and noun IDs and their combinations, but we did not observe any improvement so we omit them from the feature set. The bottom part of Table 2 summarizes the additional feature set, some of which are described below in more detail.

<sup>3</sup>There is a slight complication here. For each noun  $n$ , EM clustering estimates a probability distribution  $P(n|c^*)$  for  $n$  and semantic class  $c^*$ . From this distribution we obtained discrete semantic word classes by assigning each noun  $n$  to semantic class  $c = \operatorname{argmax}_{c^*} p(c^*|n)$ .

<sup>4</sup><http://www.post.japanpost.jp/zipcode/download.html>

As for TR and EX, our intuition is that if a problem nucleus and an aid nucleus are an adequate match, their excitation polarities are opposite, as described in Section 2. We encode whether the excitation polarities of nuclei templates are the same or not in our features. Also, the excitation polarities of problem and aid nuclei and TR are combined (TRES1, TRES2) so that the classifier can know whether the nuclei follow the constraint for adequate matches described in Section 2.

As for SIM, if an aid message matches a problem report, besides the common nucleus noun, it is reasonable to assume that certain contexts are semantically similar. We capture this characteristic in three ways. SIM1 looks for common semantic word classes in the problem report and aid message. SIM2 and SIM3 target the modifiers of the common nucleus noun if they exist.

We also observed that if an aid message matches a problem report, the problem nucleus template and aid nucleus template are often distributionally similar. A typical example is “X is sold out” and “buy X”. SIM4 captures this tendency. As the distributional similarity between templates, we used a Bayesian distributional similarity measure proposed by Kazama et al. (2010).<sup>5</sup>

CTP indicates whether the problem and aid nuclei are in contradiction relation or not. This feature was implemented based on the observation that when problem and aid nuclei are in contradiction relation, they are often proper matches (e.g., ⟨blackout, “X starts”⟩ and ⟨blackout, “X ends”⟩). CTP indicates whether nucleus pairs are in the one million contradiction phrase pairs<sup>6</sup> automatically obtained by applying a method proposed by Hashimoto et al. (2012) to 600 million Web pages.

## 5 Experiments

We evaluated our problem report recognizer and problem-aid match recognizer. For the sake of space, we give only the performance figures of the aid message recognizer at the end of Section 5.1.

We collected tweets posted during and after the 2011 Great East Japan Earthquake, between March 10 and April 4, 2011. After applying keyword-based filtering with a list of over 300

<sup>5</sup>The original similarity was defined over noun pairs and it was estimated from dependency relations. Obtaining similarity between template pairs, not noun pairs, is straightforward given the same dependency relations. We used 600 million Web pages for this similarity estimation.

<sup>6</sup>The precision of the pairs was reported as around 70%.

disaster related keywords, we obtained 55 million tweets. After dependency parsing<sup>7</sup>, we used them in our evaluation.

### 5.1 Problem Report Recognition

Firstly, we evaluated our problem report recognizer. Particularly, we assessed the effect of excitation polarities and trouble expressions in two settings. The first is against a naturally distributed gold standard data. The second targets problem reports with problem nuclei unseen in the training data.

In both experiments we observed that the performance drops when excitation polarities and trouble expressions are removed from the feature set. The performance drop was larger in the second experiment which suggests that the excitation polarities and trouble expressions are more effective against unseen problem reports.

Training and test data for problem report recognition consist of tweet-nucleus candidate pairs randomly sampled from our 55 million tweet data. The training data ( $R$ ) and test data ( $T$ ) consist of 13,000 and 1,000 pairs, respectively, manually labeled by three annotators (other than the authors) as problem or other. Final judgment was made by majority vote. The Fleiss’ kappa score for training and test data for annotation judgement is 0.74 (substantial agreement).

Our problem report recognizer and its variants are listed in Table 3. Table 4 shows the evaluation results. The proposed method achieved about 44% recall and nearly 80% precision, outperforming all other systems in terms of precision, F-score and average precision<sup>8</sup>. The improvement in precision when using TR&EX is statistically significant ( $p < 0.05$ ).<sup>9</sup> Note that F-measure dropped

<p><b>PROPOSED:</b> Our proposed method with all features used.</p> <p><b>PROPOSED-*</b>: The proposed method without the feature set denoted by “*”. Here EX and TR denote all excitation polarity and trouble expression related features, respectively, including their combinations (TRES1).</p> <p><b>PROPOSED+OET:</b> The proposed method incorporating the classification results of problem nucleus candidates by the Opinion Extraction Tool as additional binary features.</p> <p><b>RULE-BASED:</b> The method that regards only nuclei satisfying the constraint in Table 1 as problem nuclei.</p>
---

Table 3: Evaluated problem report recognizers.

<sup>7</sup><http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP>

<sup>8</sup>We calculate average precision using the formula:  $AP = \frac{\sum_{k=1}^n (Prec(k) \times rel(k))}{n}$ , where  $Prec(k)$  is the precision at cut-off  $k$  and  $rel(k)$  is an indicator function equaling 1 if the item at rank  $k$  is relevant, zero otherwise.

<sup>9</sup>Throughout this paper we performed two-tailed test of

Recognition system	R (%)	P (%)	F (%)	aP (%)
PROPOSED	44.26	<b>79.41</b>	<b>56.83</b>	<b>71.82</b>
PROPOSED-TR&EX	<b>45.08</b>	74.83	56.26	69.67
PROPOSED-EX	44.67	74.66	55.89	69.90
PROPOSED-TR	43.85	74.31	55.15	69.44
PROPOSED-MSA	28.69	70.71	40.81	57.74
PROPOSED-SWC	43.42	75.97	55.25	70.61
PROPOSED-WSP	43.14	77.83	55.50	70.45
PROPOSED-REQ	42.64	76.16	55.50	54.67
PROPOSED-GL	44.14	78.34	55.50	56.46
PROPOSED+OET	44.24	<b>79.41</b>	56.82	71.81
RULE-BASED	30.32	67.96	41.93	n/a

Table 4: Recall (R), precision (P), F-score (F) and average precision (aP) of the problem report recognizers.

whenever each type of feature was removed, implying that each type of feature is effective in this task. Especially note the performance drop if we remove excitation polarities (EX), trouble expression (TR) and both excitation and trouble expression features (TR&EX), confirming that they are crucial in recognizing problem reports with high accuracy. Also note that the performance of PROPOSED+OET was actually slightly worse than that of the proposed method. This suggests that full-fledged sentiment analysis is not effective at least in this setting. The rule-based method achieved relatively high precision despite of the low recall, demonstrating the importance of problem and aid nuclei formulations described in Section 1.

The second experiment assessed the efficiency of our problem report recognizer against unseen problem nuclei under the condition that every template in nuclei has excitation polarity. We sampled the training and test data so that the problem nucleus nouns and templates in the training and test data are disjoint. First we created a subset of the test data by selecting the samples which had nuclei with excitation templates. We call this subset  $T'$ . Next, we removed samples from training data  $R$  if either of their problem nouns or templates appeared in the nuclei of  $T'$ . The resulting new training data (called  $R'$ ) and test data ( $T'$ ) consist of 6,484 and 407 tweet-nucleus candidate pairs, respectively. We trained our problem report recognizer using  $R'$  and tested its performance using  $T'$ . Figure 2 shows the precision-recall curves obtained by changing the threshold on the SVM scores. The effectiveness of excitation polarities and trouble expressions was more evident in this setting. The PROPOSED’s performance was actually better in this setting (almost 50% recall at

population proportion (Ott and Longnecker, 2010) using SVM-threshold=0.

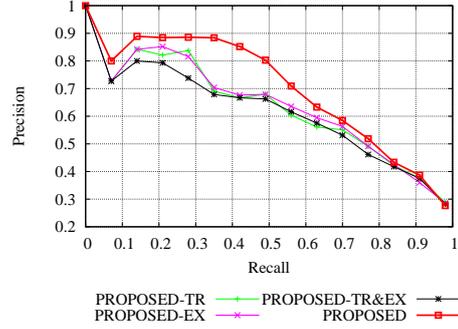


Figure 2: Precision-recall curves of problem report recognizers against unseen problem nuclei.

more than 80% precision), than the previous setting, showing that excitation templates and trouble expressions are crucial in achieving high performance especially for unseen problem nuclei. The same was confirmed when we removed excitation polarity and trouble expression related features, with performance dropping by 7.43 points in terms of average precision. The improvement in precision when using TR&EX is statistically significant ( $p < 0.01$ ). This implies, assuming that we have a wide-coverage dictionary of templates with excitation polarities, that excitation polarities are important in dealing with unexpected problems in disaster situations.

We also evaluated the aid-message recognizer, using tweet-nucleus pairs in  $R$  and  $T$  as training and test data and the annotation scheme was also the same. The average Fleiss’ kappa score was 0.55 (moderate agreement). Our recognizer achieved 53.82% recall and 65.67% precision and showed similar tendencies with the problem report recognizer, with the excitation polarities and trouble expressions contributing to higher accuracy.

We can conclude that excitation polarities and trouble expressions are important in identifying problem reports and aid messages during disaster situations.

## 5.2 Problem-Aid Matching

Next, we evaluated the performance of the problem-aid match recognizer. We applied our problem report recognizer and aid message recognizer to all 55 million tweets and combined the tweet-nucleus pairs judged as problem reports and aid messages, respectively, to create the training and test data.

The training data consists of two parts ( $M1$  and  $M2$ ).  $M1$  includes many variations of the aid messages for each problem report, while  $M2$  en-

sure diversity in nouns and templates in problem nuclei. For  $M1$ , we randomly picked up problem reports from the output of the problem report recognizer and to each we attached up to 30 randomly picked, distinct aid messages that have the same nucleus noun. Building  $M2$  follows the construction method of  $M1$ , except that: (1) we used up to 30 distinct problem nuclei for each noun; (2) for each problem report we attached only *one* randomly picked aid message.

In creating the test data  $T2$ , we followed the construction method used for  $M2$  to assess the performance of our proposal with a large variety of problems.  $M1$ ,  $M2$  and  $T2$  consist of 3,000, 6,000 and 1,000 samples, respectively. The annotation was done by majority vote of three human annotators (other than the authors), the average Fleiss’ kappa-score for training and test data was 0.63 (substantial agreement).

We trained the problem-aid match recognizers of Table 5 with  $M1$  and  $M2$ . The evaluation results performed on  $T2$  are shown in Table 6. We can observe that, among the nuclei related features, the trouble expression (TR) and excitation polarity (EX) features and their combination (TR&EX) contribute most to the performance, although the contribution of nuclei related features is less in comparison to the problem report and aid message recognition. The improvement in precision when using TR&EX is marginally significant ( $p = 0.056$ ). Instead, morphological and syntactic analysis (MSA) and semantic word class (SWC) features greatly improved performance.

As the final experiments, we evaluated top-ranking matches of our problem-aid match recognizer, where the recognizer classified all the possible combinations of tweet-nuclei pairs taken from 55 million tweets. In addition, we assessed the effectiveness of excitation polarities and trouble expressions by comparing all positive matches produced by our full problem-aid match recognizer (PROPOSED) and those produced by the problem-aid match recognizer (PROPOSED-TR&EX) that

<b>PROPOSED:</b> Our proposed method with all features used.
<b>PROPOSED-*</b> : The proposed method without the feature set denoted by “*”. Here also EX and TR denote all excitation polarity and trouble expression related features, respectively, including their combinations (TRES1 and TRES2).
<b>RULE-BASED:</b> The method that judges only problem-aid nuclei combinations with opposite excitation polarities as proper matches.

Table 5: Evaluated problem-aid match recognizers.

Matching system	R (%)	P (%)	F (%)	aP (%)
PROPOSED	30.67	<b>70.42</b>	<b>42.92</b>	<b>55.16</b>
PROPOSED-TR&EX	28.83	67.14	40.33	53.99
PROPOSED-EX	<b>31.29</b>	67.11	42.68	54.19
PROPOSED-TR	30.56	69.33	42.42	54.85
PROPOSED-MSA	13.50	53.66	21.57	44.52
PROPOSED-SWC	26.99	67.69	38.59	52.23
PROPOSED-WSP	30.61	69.51	42.50	54.81
PROPOSED-CTP	30.06	70.00	42.05	54.94
PROPOSED-SIM	29.95	70.11	41.97	54.98
PROPOSED-REQ	30.58	70.25	42.61	54.67
PROPOSED-GL	30.61	70.31	42.65	55.02
PROPOSED-SSR	30.67	69.44	42.72	54.91
RULE-BASED	15.33	17.36	16.28	n/a

Table 6: Recall (R), precision (P), F-score (F) and average precision (aP) of the problem-aid match recognizers.

did not use excitation polarities and trouble expressions in its feature set. Note that PROPOSED-TR&EX was fed by the problem report and aid message recognizers that didn’t use excitation polarities and trouble expressions. For both systems’ training data we used  $R$  for the problem report and aid message recognizers;  $M1$  and  $M2$  for the problem-aid matching recognizers.

PROPOSED and PROPOSED-TR&EX output 15.2 million and 13.4 million positive matches, covering 1,691 and 1,442 nucleus nouns, respectively. Table 7 shows match samples identified with PROPOSED. We observed that the output of each system was dominated by just a handful of frequent nucleus nouns, such as “water” or “gasoline”. We preferred to assess the performance of our system against a large variation of problem-aid nuclei, thus we restricted the number of matches to 10 for each noun<sup>10</sup>. After this restriction the number of matches found by PROPOSED and PROPOSED-

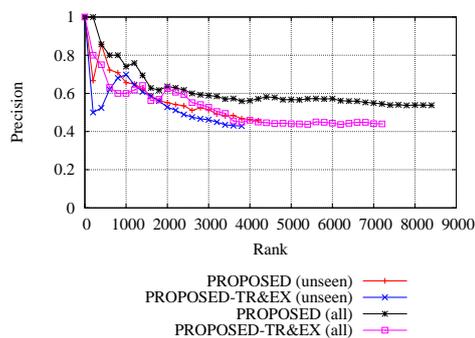


Figure 3: Problem-aid match recognition performance for ‘all’ and ‘unseen’ problem reports.

<sup>10</sup>Note that this setting is a pessimistic estimation of our system’s overall performance, since according to our observations problem reports with very frequent nucleus nouns had proper matches with a higher accuracy than problem reports with less frequent nucleus nouns.

<p><b>Problem report:</b> いわきの常磐病院、いわき泌尿器科病院、竹林貞吉記念クリニック、泉中央クリニックは、17日から透析を中止します。患者の方は至急連絡してください。(Starting from the 17th, the Iwaki Joban Hospital, the Iwaki Urology Clinic, the Takebayashi Sadakichi Memorial Clinic and the Izumi Central Clinic have all <b>suspended dialysis sessions</b>. Patients are advised to urgently make contact.)</p>
<p><b>Aid message:</b> いわき泌尿器科病院で短時間透析が可能です。受付時間は9時から16時までです。(透析の再開) (Restart of dialysis sessions: short <b>dialysis sessions are available</b> at the Iwaki Urology Clinic between 9 AM and 4 PM.)</p>
<p><b>Problem report:</b> ごめんなさい拡散をお願いしてもいいですか。仙台の父親の話ですと携帯の充電がもうない人が続出しているそうです。携帯充電器の支援が必要かと思われます。 (Please spread this message. According to my father in Sendai, there are more and more people whose phones <b>ran out of battery</b>. We need phone chargers!)</p>
<p><b>Aid message:</b> 【拡散希望】仙台若林区役所で携帯電話の充電ができるそうです。 ([Please spread] At the City Hall of Wakabayashi-ku, Sendai, you <b>can recharge</b> your phone battery.)</p>

Table 7: Examples from the output of the proposed method in the ‘all’ setting. Problem report and aid message nuclei are boldfaced in the English translations.

TR&EX was 8,484 and 7,363, respectively.

The performance of PROPOSED and PROPOSED-TR&EX were assessed in two settings: ‘all’ and ‘unseen’. For ‘all’, we selected 400 problem-aid matches from the outputs of the respective systems after applying the 10-match restriction. For ‘unseen’, first we removed the samples from the systems’ outputs if either the nucleus noun or template pair appear in the nuclei of the problem-aid match recognizers’ training data. Next we applied the same sampling process as with ‘all’. Three annotators (other than the authors) manually labeled the sample sets, final judgment being made by majority vote. The Fleiss’ kappa score for all test data was 0.73 (substantial agreement).

Figure 3 shows the systems’ precision curves, drawn from the samples whose X-axis positions represent the ranks according to SVM scores. In both scenarios we can confirm that excitation polarity and trouble expression related features contribute to this task. In the ‘all’ setting in terms of average precision calculated over the top 7,200 matches, PROPOSED’s 62.36% is 10.48 points higher than that of PROPOSED-TR&EX. For unseen problem/aid nuclei PROPOSED method’s average precision of 58.57% calculated at the top 3,800 matches is 5.47 points higher than that of PROPOSED-TR&EX at the same data point. The improvement in precision when using TR&EX is

statistically significant in both settings ( $p < 0.01$ ).

## 6 Related Work

Twitter has been observed as a platform for situational awareness during various crisis situations (Starbird et al., 2010; Vieweg et al., 2010), as sensors for an earthquake reporting system (Sakaki et al., 2010; Okazaki and Matsuo, 2010) or to detect epidemics (Aramaki et al., 2011). Besides Twitter, blogs or forums have also been the target of community response analysis (Qu et al., 2009; Torrey et al., 2007). Similar to our work are the ones of Neubig et al. (2011) and Ishino et al. (2012), who tackle specific problems that occur during disasters (i.e., *safety information* and *transportation information*, respectively); and Munro (2011), who extracted “actionable messages” (requests and aids, indiscriminately), matching being performed manually. Our work differs from (Neubig et al., 2011) and (Ishino et al., 2012) in that we do not restrict the range of problem reports, and as opposed to (Munro, 2011), matching is automatic.

Systems such as that of Seki (2011)<sup>11</sup> or Munro (2013)<sup>12</sup> are successful examples of crisis crowdsourcing, but these require extensive human intervention to coordinate useful information.

Another category of related work relevant to our task is troubleshooting. Baldwin et al. (2007) and Raghavan et al. (2010) use discussion forums to solve technical problems using supervised learning methods, but these approaches presume that the solution of a specific problem is within the same thread. In our work we do not employ structural characteristics of tweets as restrictions (e.g., a problem report and its aid message need to be in the same tweet chain).

## 7 Conclusions

In this paper, we proposed a method to discover matches between problem reports and aid messages from tweets in large-scale disasters. Through a series of experiments, we demonstrated that the performance of the problem-aid matching can be improved with the usage of semantic orientation of excitation polarities, proposed in (Hashimoto et al., 2012), and trouble expressions.

We are planning to deploy our system and release model files of the classifiers to assist relief efforts in future crisis scenarios.

<sup>11</sup><http://www.sinsai.info/>

<sup>12</sup><http://www.mission4636.org/>

## References

- Adam Acar and Yuya Muraki. 2011. Twitter for crisis communication: Lessons learned from Japan's tsunami disaster. *International Journal of Web Based Communities*, 7(3):392–402.
- Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita. 2011. Twitter catches the flu: Detecting influenza epidemics using Twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1568–1576.
- Timothy Baldwin, David Martinez, and Richard B. Penman. 2007. Automatic thread classification for Linux user forum information access. In *Proceedings of the 12th Australasian Document Computing Symposium (ADCS 2007)*, pages 72–79.
- Stijn De Saeger, Kentaro Torisawa, and Jun'ichi Kazama. 2008. Looking for trouble. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 185–192.
- Stijn De Saeger, Kentaro Torisawa, Masaaki Tsuchida, Jun'ichi Kazama, Chikara Hashimoto, Ichiro Yamada, Jong-Hoon Oh, István Varga, and Yulan Yan. 2011. Relation acquisition using word classes and partial patterns. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 825–835.
- Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 5:378–382.
- Chikara Hashimoto, Kentaro Torisawa, Stijn De Saeger, Jong-Hoon Oh, and Jun'ichi Kazama. 2012. Excitatory or inhibitory: A new semantic orientation extracts contradiction and causality from the web. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 619–630.
- Aya Ishino, Shuhei Odawara, Hidetsugu Nanba, and Toshiyuki Takezawa. 2012. Extracting transportation information and traffic problems from tweets during a disaster: Where do you evacuate to? In *Proceedings of the Second International Conference on Advances in Information Mining and Management (IMMM 2012)*, pages 91–96.
- Hiroshi Kanayama and Tetsuya Nasukawa. 2008. Textual demand analysis: Detection of users' wants and needs from opinions. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 409–416.
- Jun'ichi Kazama and Kentaro Torisawa. 2008. Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, pages 407–415.
- Jun'ichi Kazama, Stijn De Saeger, Kow Kuroda, Masaki Murata, and Kentaro Torisawa. 2010. A Bayesian method for robust estimation of distributional similarities. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 247–256.
- Jun'ichi Kazama, Stijn De Saeger, Kentaro Torisawa, Jun Goto, and István Varga. 2013. Saigaiji jouhou e no shitsumon outo shisutemu no tekiyou no kokoromi. (An attempt for applying question-answering system on disaster related information). In *Proceeding of the Nineteenth Annual Meeting of The Association for Natural Language Processing*. (in Japanese).
- Benjamin Mandel, Aron Culotta, John Boulahanis, Danielle Stark, Bonnie Lewis, and Jeremy Rodrigue. 2012. A demographic analysis of online sentiment during Hurricane Irene. In *Proceedings of the Second Workshop on Language Analysis in Social Media (LASM 2012)*, pages 27–36.
- Robert Munro. 2011. Subword and spatiotemporal models for identifying actionable information in Haitian Kreyol. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL-2011)*, pages 68–77.
- Robert Munro. 2013. Crowdsourcing and the crisis-affected community. *Information Retrieval*, 16(2):210–266.
- Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using CRFs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 786–794.
- National Police Agency of Japan. 2013. Damage situation and public countermeasures associated with 2011 Tohoku district – off the Pacific Ocean Earthquake. [http://www.npa.go.jp/archive/keibi/biki/higaijokyo\\_e.pdf](http://www.npa.go.jp/archive/keibi/biki/higaijokyo_e.pdf). (accessed on 30 April, 2013).
- Graham Neubig, Yuichiroh Matsubayashi, Masato Hagiwara, and Koji Murakami. 2011. Safety information mining what can NLP do in a disaster . In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 965–973.
- Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Takuya Kawada, Stijn De Saeger, Jun'ichi Kazama, and Yiou Wang. 2012. Why question answering using sentiment analysis and word classes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 368–378.

- Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Motoki Sano, Stijn De Saeger, and Kiyonori Ohtake. 2013. Why-question answering using intra- and inter-sentential causal relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*.
- Kiyonori Ohtake, Kentaro Torisawa, Jun Goto, and Stijn De Saeger. 2013. Saigaiji ni okeru hisaisha to kyūen kyūujōsha kan no sōhōko komyūnikēshon. (Bi-directional communication between victims and rescuers during a crisis). In *Proceeding of the Nineteenth Annual Meeting of The Association for Natural Language Processing*. (in Japanese).
- Makoto Okazaki and Yutaka Matsuo. 2010. Semantic Twitter: Analyzing tweets for real-time event notification. In *Proceedings of the 2008/2009 international conference on Social software: Recent trends and developments in social software (BlogTalk 2008)*, pages 63–74.
- R. Lyman Ott and Michael T. Longnecker, 2010. *An Introduction to Statistical Methods and Data Analysis*, chapter 10.2. Brooks Cole, 6th edition.
- Yan Qu, Philip Fei Wu, and Xiaoqing Wang. 2009. Online community response to major disaster: A study of Tianya forum in the 2008 Sichuan Earthquake. In *42st Hawaii International International Conference on Systems Science (HICSS-42)*, pages 1–11.
- Preethi Raghavan, Rose Catherine, Shajith Ikkal, Nanda Kambhatla, and Debapriyo Majumdar. 2010. Extracting problem and resolution information from online discussion forums. In *Proceedings of the 16th International Conference on Management of Data (COMAD 2010)*.
- Takeo Saijo. 2012. *Hito-o tasukeru sungoi shikumi. (A stunning system that saves people)*. Diamond Inc. (in Japanese).
- Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW 2010)*, pages 851–860.
- Motoki Sano, István Varga, Jun'ichi Kazama, and Kentaro Torisawa. 2012. Requests in tweets during a crisis: A systemic functional analysis of tweets on the Great East Japan Earthquake and the Fukushima Daiichi nuclear disaster. In *Papers from the 39th International Systemic Functional Congress (ISFC39)*, pages 135–140.
- Haruyuki Seki. 2011. Higashi-nihon daishinsai fukkō shien platform sinsai.info no naritachi to kōgo no kadai. (The organizational structure of sinsai.info restoration support platform for the 2011 Great East Japan Earthquake and future challenges). *Journal of digital practices*, 2(4):237–241. (in Japanese).
- Kate Starbird, Leysia Palen, Amanda L. Hughes, and Sarah Vieweg. 2010. Chatter on the red: What hazards threat reveals about the social life of microblogged information. In *Proceedings of The 2010 ACM Conference on Computer Supported Cooperative Work (CSCW 2010)*, pages 241–250.
- Cristen Torrey, Moira Burke, Matthew L. Lee, Anind K. Dey, Susan R. Fussell, and Sara B. Kiesler. 2007. Connected giving: Ordinary people coordinating disaster relief on the Internet. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS-40)*, pages 179–188.
- Katerina Tsagkalidou, Vassiliki Koutsonikola, Athena Vakali, and Konstantinos Kafetsios. 2011. Emotional aware clustering on micro-blogging sources. In *Proceedings of the 4th international conference on Affective computing and intelligent interaction (ACII 2011)*, pages 387–396.
- Sarah Vieweg, Amanda L. Hughes, Kate Starbird, and Leysia Palen. 2010. Microblogging during two natural hazards events: What Twitter may contribute to situational awareness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2010)*, pages 1079–1088.
- Patrick Winn. 2011. Japan tsunami disaster: As Japan scrambles, Twitter reigns. *GlobalPost*, 18 March.

# A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations

Angeliki Lazaridou

University of Trento

angeliki.lazaridou@unitn.it

Ivan Titov

Saarland University

titov@mmci.uni-saarland.de

Caroline Sporleder

Trier University

csporled@coli.uni-sb.de

## Abstract

We propose a joint model for unsupervised induction of sentiment, aspect and discourse information and show that by incorporating a notion of latent discourse relations in the model, we improve the prediction accuracy for aspect and sentiment polarity on the sub-sentential level. We deviate from the traditional view of discourse, as we induce types of discourse relations and associated discourse cues relevant to the considered opinion analysis task; consequently, the induced discourse relations play the role of opinion and aspect shifters. The quantitative analysis that we conducted indicated that the integration of a discourse model increased the prediction accuracy results with respect to the discourse-agnostic approach and the qualitative analysis suggests that the induced representations encode a meaningful discourse structure.

## 1 Introduction

With the rapid growth of the Web, it is becoming increasingly difficult to discern useful from irrelevant information, particularly in user-generated content, such as product reviews. To make it easier for the reader to separate the wheat from the chaff, it is necessary to structure the available information. In the review domain, this is done in *aspect-based sentiment analysis* which aims at identifying text fragments in which *opinions* are expressed about ratable *aspects of products*, such as ‘room quality’ or ‘service quality’. Such fine-grained analysis can serve as the first step in *aspect-based sentiment summarization* (Hu and Liu, 2004), a task with many practical applications.

Aspect-based summarization is an active research area for which various techniques have been developed, both statistical (Mei et al., 2007; Titov and McDonald, 2008b) and not (Hu and Liu, 2004), and relying on different types of supervision sources, such as sentiment-annotated texts or polarity lexica (Turney and Littman, 2002). Most methods rely on local information (bag-of-words, short ngrams or elementary syntactic fragments) and do not attempt to account for more complex interactions. However, these local lexical representations by themselves are often not sufficient to infer a sentiment or aspect for a fragment of text. For instance, in the following example taken from a TripAdvisor<sup>1</sup> review:

**Example 1.** *The room was nice but let’s not talk about the view.*

it is difficult to deduce on the basis of local lexical features alone that the opinion about the view is negative. The clause *let’s not talk about the view* could by itself be neutral or even positive given the right context (e.g., *I’ve never seen such a fancy hotel room, my living room doesn’t look that cool... and let’s not talk about the view*). However, the contrast relation signaled by the connective *but* makes it clear that the second clause has a negative polarity. The same observations can be made about transitions between aspects: changes in aspect are often clearly marked by discourse connectives. Importantly, some of these cues are not discourse connectives in the strict linguistic sense and are specific to the review domain (e.g., the phrase *I would also* in a review indicates that the topic is likely to be changed). In order to accurately predict sentiment and topic,<sup>2</sup> a model needs to ac-

<sup>1</sup><http://www.tripadvisor.com/>

<sup>2</sup>In what follows, we use the terms *aspect* and *topic*, inter-

count for these discourse phenomena and cannot rely solely on local lexical information.

These issues have not gone unnoticed to the research community. Consequently, there has recently been an increased interest in models that leverage content and discourse structure in sentiment analysis tasks. However, discourse-level information is typically incorporated in a pipeline architecture, either in the form of sentiment polarity shifters (Polanyi and Zaenen, 2006; Nakagawa et al., 2010) that operate on the lexical level or by using discourse relations (Taboada et al., 2008; Zhou et al., 2011) that comply with discourse theories like Rhetorical Structure Theory (RST) (Mann and Thompson, 1988). Such approaches have a number of disadvantages. First, they require additional resources, such as lists of polarity shifters or discourse connectives which signal specific relations. These resources are available only for a handful of languages. Second, relying on a generic discourse analysis step that is carried out before sentiment analysis may introduce additional noise and lead to error propagation. Furthermore, these techniques will not necessarily be able to induce discourse relations informative for the sentiment analysis domain (Voll and Taboada, 2007).

An alternative approach is to define a task-specific scheme of discourse relations (Somasundaran et al., 2009). This previous work showed that task-specific discourse relations are helpful in predicting sentiment, however, in doing so they relied on gold-standard discourse annotation at test time rather than predicting it automatically or inducing it jointly with sentiment polarity.

We take a different approach and induce discourse and sentiment information jointly in an unsupervised (or weakly supervised) manner. This has the advantage of not having to pre-specify a mapping from discourse cues to discourse relations; our model induces this automatically, which makes it portable to new domains and languages. Joint induction of discourse and sentiment structure also has the added benefit that the model is able to learn exactly those aspects of discourse structure that are relevant for sentiment analysis.

We start with a relatively standard joint model of sentiment and topic, which can be regarded as a cross-breed between the JST model (Lin and He, 2009) and the ASUM model (Jo and Oh, 2011),

changeably as well as *sentiment levels* and *opinion polarity*.

both state-of-the-art techniques. This model is weakly supervised, as it relies solely on document-level (i.e. not aspect-specific) opinion polarity labels to induce topics and sentiment on the sub-sentential level. In order to test our hypothesis that discourse information is beneficial, we add a discourse modeling component. Note that in modeling discourse we do not exploit any kind of supervision. We demonstrate that the resulting model outperforms the baseline on a product review dataset (see Section 5).

To the best of our knowledge, unsupervised joint induction of discourse structure, sentiment and topic information has not been considered before, particularly not in the context of the aspect-based sentiment analysis task. Importantly, our method for discourse modeling is a general method which can be integrated in virtually any LDA-style model of aspect and sentiment.

## 2 Modeling Discourse Structure

Discourse cues typically do not directly indicate sentiment polarity (or aspect). However, they can indicate how polarity (or aspect) changes as the text unfolds. As we have seen in the examples above, changes in polarity can happen on a sub-sentential level, i.e., between adjacent clauses or, from a discourse-theoretic point of view, between adjacent *elementary discourse units (EDUs)*. To model these changes we need a strong linguistic signal, for example, in the form of discourse connectives or other discourse cues. We hypothesize that these are more likely to occur at the beginning of an EDU than in the middle or at the end. This is certainly true for most of the traditional discourse relation cues (particularly connectives).

Changes in polarity or aspect are often correlated with specific discourse relations, such as ‘contrast’. However, not all relations are relevant and there is no one-to-one correspondence between relations and sentiment changes.<sup>3</sup> Furthermore, if a discourse relation signals a change, it is typically ambiguous whether this change occurs with the polarity (example 1) or the aspect (*the room was nice but the breakfast was even better*) or both (*the room was nice but the breakfast was awful*). Therefore, we do not explicitly model

<sup>3</sup>The ‘explanation’ relation, for example, can occur with a polarity change (*We were upgraded to a really nice room because the hotel made a terrible blunder with our booking*) but does not have to (*The room was really nice because the hotel was newly renovated*).

Name	Description
AltSame	<b>different</b> polarity, <b>same</b> aspect
SameAlt	<b>same</b> polarity, <b>different</b> aspect
AltAlt	<b>different</b> polarity and aspect

Table 1: Discourse relations

generic discourse relations; instead, inspired by the work of Somasundaran et al. (2008), we define three very general relations which encode how polarity and aspect change (Table 1). Note that we do not have a discourse relation *SameSame* since we do not expect to have strong linguistic evidence which states that an EDU contains the same sentiment information as the previous one.<sup>4</sup> However, we assume that the sentiment and topic flow is fairly smooth in general. In other words, for two adjacent EDUs not connected by any of the above three relations, the prior probability of staying at the same topic and sentiment level is higher than picking a new topic and sentiment level (i.e. we use “sticky states” (Fox et al., 2008)).

### 3 Model

In this section we describe our Bayesian model, first the discourse-agnostic model and then an extension needed to encode discourse information. The formal generative story is presented in Figure 1: the red fragments correspond to the discourse modeling component. In order to obtain the generative story for the discourse-agnostic model, they simply need to be ignored.

#### 3.1 Discourse-agnostic model

In our approach we make an assumption that all the words in an EDU correspond to the same topic and sentiment level. We also assume that an overall sentiment of the document is defined, this is the only supervision we use in inducing the model. Unlike some of the previous work (e.g., (Titov and McDonald, 2008a)), we do not constrain aspect-specific sentiment to be the same across the document. We describe our discourse-agnostic model by first describing the set of corpus-level and document-level parameters, and then explain how the content of each document is generated.

**Drawing model parameters** On the corpus level, for every topic  $z \in \{1, \dots, K\}$  and every sentiment polarity level  $y \in \{-1, 0, +1\}$ , we start by drawing a unigram language model

<sup>4</sup>The typical connective in this situation would be *and* which is highly ambiguous and can signal several traditional discourse relations.

from a Dirichlet prior. For example, the language model of the aspect *service* may indicate that the word *friendly* is used to express a positive opinion, whereas the word *rude* expresses a negative one.

Similarly, for every topic  $z$  and every overall sentiment polarity  $\hat{y}$ , we draw a distribution  $\psi_{\hat{y},z}$  over opinion polarity in this topic  $z$ . Intuitively, one would expect the sentiment of an aspect to more often agree with the overall sentiment  $\hat{y}$  than not. This intuition is encoded in an asymmetric Dirichlet prior  $Dir(\gamma_{\hat{y}})$  for  $\psi_{\hat{y},z} : \gamma_{\hat{y}} = (\gamma_{\hat{y},1}, \dots, \gamma_{\hat{y},M})$ ,  $\gamma_{\hat{y},y} = \beta + \tau\delta_{y,\hat{y}}$ , where  $\delta_{y,\hat{y}}$  is the Kronecker symbol,  $\beta$  and  $\tau$  are nonnegative scalar parameters. Using these “heavy-diagonal” priors is crucial, as this is the way to ensure that the overall sentiment level is tied to the aspect-specific sentiment level. Otherwise, sentiment levels will be specific to individual aspects (e.g., the “+1” sentiment for one topic may correspond to a “-1” sentiment for another one). Without this property we would not be able to encode soft constraints imposed by the discourse relations.

**Drawing documents** On the document level, as in the standard LDA model, we choose the distribution over topics for the document from a symmetric Dirichlet prior parametrized by  $\alpha$ , which is used to control sparsity of topic assignments. Furthermore, we draw the global sentiment  $\hat{y}_d$  from a uniform distribution.

The generation of a document is done on the EDU-by-EDU basis. In this work, we assume that EDU segmentation is provided by the preprocessing step. First, we generate the aspect  $z_{d,s}$  for EDU  $s$  according to the distribution of topics  $\theta_d$ . Then, we choose a sentiment level  $y_{d,s}$  for the considered EDU from the categorical distribution  $\psi_{\hat{y}_d, z_{d,s}}$ , conditioned on the aspect  $z_{d,s}$ , as well as on the global sentiment of the document  $\hat{y}_d$ . Finally, we generate the bag of words for the EDU by drawing the words from the aspect- and sentiment-specific language model.

This model can be seen as a variant of a state-of-the-art model for jointly inducing sentiment and aspect at the sentence level (Jo and Oh, 2011), or, more precisely, as its combination with the JST model (Lin and He, 2009), adapted to the specifics of our setting. Both these models have been shown to perform well on sentiment and topic prediction tasks, outperforming earlier models, such as the TSM model (Mei et al., 2007). Consequently, it can be considered as a strong baseline.

### 3.2 Discourse-informed model

In order to integrate discourse information into the discourse-agnostic model, we need to define a set of extra parameters and random variables.

**Drawing model parameters** First, at the corpus level, we draw a distribution  $\tilde{\varphi}$  over four discourse relations: three relations as defined in Table 1 and an additional dummy relation 4 to indicate that there is no relation between two adjacent EDUs (*NoRelation*). This distribution is drawn from an asymmetric Dirichlet prior parametrized by a vector of hyperparameters  $\nu$ . These parameters encode the intuition that most pairs of EDUs do not exhibit a discourse relation relevant for the task (i.e. favor *NoRelation*), that is  $\nu_4$  has a distinct and larger value than other parameters  $\nu_{\bar{4}}$ .

Every discourse relation  $c$  (including *NoRelation* which is treated here as *Same-Same*) is associated with two groups of transition distributions, one governing transitions of sentiment ( $\tilde{\psi}_c$ ) and another one controlling topic transitions ( $\tilde{\theta}_c$ ). The parameter  $\tilde{\psi}_{c,y_s}$ , defines a distribution over sentiment polarity for the EDU  $s + 1$  given the sentiment for the  $s$ th EDU  $y_s$  and the discourse relation  $c$ . This distribution encodes our beliefs about sentiment transitions between EDUs  $s$  and  $s + 1$  related through  $c$ . For example, the distribution  $\tilde{\psi}_{SameAlt,+1}$  would assign higher probability mass to the positive sentiment polarity (+1) than to the other 2 sentiment levels (0, -1). Similarly, the parameter  $\tilde{\theta}_{c,z_s}$ , defines a distribution over  $K$  aspects.

These two families of transition distributions are each defined in the following way. For the distribution  $\tilde{\theta}$ , for relations that favor changing the aspect (*SameAlt* and *AltAlt*), the probability of the preferred ( $K-1$ ) transitions is proportional to  $\omega_\theta$  and for the remaining transitions it is proportional to 1. On the other hand, for the relations that favor keeping the same aspect (*NoRelation* and *AltSame*), the probability of the preferred transition is proportional to  $\omega'_\theta$ , whereas the probability of the ( $K-1$ ) remaining transitions is again proportional to 1. For the sentiment transitions, the distribution  $\tilde{\psi}_{c,y_s}$  is defined in the analogous way (but depends on  $\omega_\psi$  and  $\omega'_\psi$ ). These scalars are hand-coded and define soft constraints that discourse relations impose on the local flow of sentiment and aspects.

The parameter  $\tilde{\phi}_c$  is a language model over discourse cues  $\tilde{w}$ , which are not restricted to unigrams but can generate phrases of arbitrary (and

variable) size. For this reason, we draw them from a Dirichlet process (DP) (i.e. one for each discourse relation, except for *NoRelation*). The base measure  $G_0$  provides the probability of an  $n$ -word sequence calculated with the bigram probability model estimated from the corpus.<sup>5</sup> This model component bears strong similarities to the Bayesian model of word segmentation (Goldwater et al., 2009), though we use the DP process to generate only the prefix of the EDU, whereas the rest of the EDU is generated from the bag-of-words model.

**Drawing documents** As pointed out above, the content generation is broken into two steps, where first we draw the discourse cue  $\tilde{w}_{d,s}$  from  $\tilde{\phi}_c$  and then we generate the remaining words.

The second difference at the data generation step (Figure 1) is in the way the aspect and sentiment labels are drawn. As the discourse relation between the EDUs has already been chosen, we have some expectations about the values of the sentiment and aspect of the following EDU, which are encoded by the distributions  $\tilde{\psi}$  and  $\tilde{\theta}$ . These are only soft constraints that have to be taken into consideration along with the information provided by the aspect-sentiment model. This coupling of information naturally translates into the *product-of-experts* (PoE) (Hinton, 1999) approach, where two sources of information jointly contribute to the final result. The PoE model seems to be more appropriate here than a mixture model, as we do not want the discourse transition to overpower the sentiment-topic model. In the PoE model, in order for an outcome to be chosen, it needs to have a non-negligible probability under both models.

## 4 Inference

Since exact inference of our model is intractable, we use collapsed Gibbs sampling. The variables that need to be inferred are the topic assignments  $\mathbf{z}$ , the sentiment assignments  $\mathbf{y}$ , the discourse relations  $\mathbf{c}$  and the discourse cue  $\tilde{w}$  (or, more precisely, its length) and are all sampled jointly (for each EDU) since we expect them to be highly dependent. All other variables (i.e. unknown distributions) could be marginalized out to obtain a collapsed Gibbs sampler (Griffiths and Steyvers, 2004).

<sup>5</sup>This measure is improper but it serves the purpose of favoring long cues, the behavior arguably desirable for our application.

Global parameters:	
$\tilde{\varphi} \sim Dir(\nu)$	[distrib of disc rel]
for each discourse relation $c = 1, \dots, 4$ :	
$\tilde{\phi}_c \sim DP(\eta, G_o)$	[distrib of disc rel specific disc cues]
$\tilde{\theta}_{c,k}$ - fixed	[distrib of rel specific aspect transitions]
$\tilde{\phi}_{c,y}$ - fixed	[distrib of rel specific sent transitions]
for each aspect $k = 1, 2 \dots K$ :	
for each sentiment $y = -1, 0, +1$ :	
$\phi_{k,y} \sim Dir(\lambda_k)$	[unigram language models]
for each global sentiment $\hat{y} = -1, 0, +1$ :	
$\psi_{\hat{y},k} \sim Dir(\gamma)$	[sent distrib given overall sentiment]
Data Generation:	
for each document $d$ :	
$\hat{y}_d \sim Unif(-1, 0, +1)$	[global sentiment]
$\theta_d \sim Dir(\alpha)$	[distr over aspects]
for every EDU $s$ :	
$c_{d,s} \sim \tilde{\varphi}$	[draw disc relation]
if $c_{d,s} \neq NoRelation$	
$\tilde{w}_{d,s} \sim \tilde{\phi}_{c_{d,s}}$	[draw disc cue]
$z_{d,s} \sim \theta_d * \tilde{\theta}_{c_{d,s}, z_{d,s}-1}$	[draw aspect]
$y_{d,s} \sim \psi_{\hat{y}_d, z_{d,s}} * \tilde{\psi}_{c_{d,s}, y_{d,s}-1}$	[draw sentiment level]
for each word after disc cue:	
$w_{d,s} \sim \phi_{z_{d,s}, y_{d,s}}$	[draw words]

Figure 1: The generative story for the joint model. The components responsible for modeling discourse information are emphasized in red: when dropped, one is left with the discourse-agnostic model.

Unfortunately, the use of the PoE model prevents us from marginalizing the parameters exactly. Instead, as in Naseem et al. (2009), we resort to an approximation. We assume that  $z_{d,s}$  and  $y_{d,s}$  are drawn twice; once from the document specific distribution and once from the discourse transition distributions. Under this simplification, we can easily derive the conditional probabilities for the collapsed Gibbs sampling.

## 5 Experiments

To the best of our knowledge, this is the first work that aims at evaluating directly the joint information of the sentiment and aspect assignment at the sub-sentential level of full reviews; most existing studies either focus on indirect evaluation of the produced models (e.g., classifying the overall sentiment of sentences (Titov and McDonald, 2008a; Brody and Elhadad, 2010) or even reviews (Nakagawa et al., 2010; Jo and Oh, 2011)) or evaluated solely at the sentential or even document level. Consequently, in order to evaluate our methods, we created a new dataset which will be publicly released.

Aspects	Frequency
service	246
value	55
location	121
rooms	316
sleep quality	56
cleanliness	59
amenities	180
food	81
recommendation	121
rest	306
<b>Total</b>	<b>1541</b>

Table 2: Distribution of aspects in the data.

**Dataset and Annotation** The dataset we created consists of 13559 hotel reviews from TripAdvisor.com.<sup>6</sup> Since our modeling is performed on the EDU level, all sentences were segmented using the SLSEG software package.<sup>7</sup> As a result, our dataset consists of 322,935 EDUs.

For creating the gold standard, 9 annotators annotated a random subset of our dataset (65 reviews, 1541 EDUs). The annotators were presented with the whole review partitioned in EDUs and were asked to annotate every EDU with the aspect and sentiment (i.e. +1, 0 or -1) it expresses. Table 2 presents the distribution of aspects in the dataset. The distribution of the sentiments is uniform. The label *rest* captures cases where EDUs do not refer to any aspect or to a very rare aspect. The inter-annotator agreement (IAA), as measured in terms of Cohen’s kappa score, was 66% for the aspect labeling, 70% for the sentiment annotation and 61% for the joint task of sentiment and aspect annotation. Though these scores may not seem very high, they are similar to the ones reported in related sentiment annotation efforts (see e.g., Ganu et al. (2009)).

**Experimental setup** In order to quantitatively evaluate the model predictions, we run two sets of experiments. In the first, we treat the task as an unsupervised classification problem and evaluate the output of the models directly against the gold standard annotation. This is a very challenging set-up, as the model has no prior information about the aspects defined (Table 2). In the second set of experiments, we show that aspects and sentiments induced by our model can be used to construct informative features for supervised classification. In

<sup>6</sup>Downloadable from <http://clic.cimec.unitn.it/~angeliki.lazaridou/datasets/ACL2013Sentiment.tar.gz>

<sup>7</sup>[www.sfu.ca/~mtaborda/research/SLSeg.html](http://www.sfu.ca/~mtaborda/research/SLSeg.html)

Model	Precision	Recall	F1
<i>Random</i>	3.9	3.8	3.8
<i>SentAsp</i>	15.0	10.2	9.2
<i>Discourse</i>	<b>16.5</b>	<b>13.8</b>	<b>10.8</b>

Table 3: Results in terms of macro-averaged precision, recall and F1.

Model	Unmarked	Marked
<i>SentAsp</i>	9.2	5.4
<i>Discourse</i>	<b>9.3</b>	<b>11.5</b>

Table 4: Separate evaluation (F1) of the “marked” and the “unmarked” EDUs.

all the cases, we compare the discourse-agnostic and the discourse-informed models.

In order to induce the model, we let the sampler run for 2000 iterations. We use the last sample to define the labeling. The number of topics  $K$  was set to 10 in order to match the number of aspects defined in our annotation scheme (see Table 2). The hyperpriors were chosen in a qualitative experiment over a subset of our dataset by manually inspecting the produced languages models. The resulting values are:  $\alpha = 10^{-3}$ ,  $\beta = 5 * 10^{-4}$ ,  $\tau = 5 * 10^{-4}$ ,  $\eta = 10^{-3}$ ,  $\nu_4 = 10^3$ ,  $\nu_{\bar{4}} = 10^{-4}$ ,  $\omega_\theta = 85$  and  $\omega'_\theta = \omega_\psi = \omega'_\psi = 5$ .

### 5.1 Direct clustering evaluation

Our labels encoding aspect and sentiment level can be regarded as clusters. Consequently we can apply techniques developed in the context of clustering evaluation. We use a version of the standard metrics considered for the word sense induction task (Agirre and Soroa, 2007) where a clustering is converted to a classification problem. This is achieved by splitting the gold standard into two subsets; the training portion is used to choose one-to-one correspondence from the gold classes to the induced clusters and then the chosen mapping is applied to the testing portion. We perform 10-fold cross validation and report precision, recall and F1 score. Our dataset is very skewed and the majority class (*rest*) is arguably the least important, so we use macro-averaging over labels and then average those across folds to arrive to the reported numbers. We compare the discourse-informed model (*Discourse*) against two baselines; the discourse-agnostic *SentAsp* model and *Random* which assigns a random label to an EDU while respecting the distribution of labels in the training set.

Table 3 presents the first analysis conducted on the full set of EDUs. We observe that by incorporating latent discourse relation we improve per-

	Content	Aspect	Polarity
1	<i>but</i> certainly off <b>its</b> greatness	value	neg
2	<i>and</i> while small <b>they</b> are nice	rooms	pos
3	<i>but</i> <b>it</b> is not free for all guests	amenities	neg
4	<i>and</i> the water was brown	clean	neg
5	<i>and</i> no tea making facilities	rooms	neg
6	<i>when</i> i checked out	service	pos
7	<i>and</i> if you do not	service	neg
8	<i>when</i> we got home	clean	neu

Table 5: Examples of EDUs where local information is not sufficiently informative.

formance over the discourse-agnostic model *SentAsp* (statistically significant according to paired t-test with  $p < 0.01$ ). Note that fairly low scores in this evaluation setting are expected for any unsupervised model of sentiment and topics, as models are unsupervised both in the aspect-specific sentiment and in topic labels and the total number of labels is 28 (all aspects can be associated with the 3 sentiment levels except for *rest* which can only be used with neutral (0) sentiment). Consequently, induced topics, though informative (as we confirm in Section 5.3), may not correspond to the topics defined in the gold standard. For example, one well-known property of LDA-style topic models is their tendency to induce topics which account for similar fraction of words in the dataset (Jagaramudi et al., 2012), thus, over-splitting ‘heavy’ topics (e.g. *rooms* in our case). The same, though to lesser degree, is true for sentiment levels where the border between neutral and positive (or negative) is also vaguely defined.

To gain insight into our model, we conducted an experiment similar to the one presented in Somasundaran et al. (2009). We divide the dataset in two subsets; one containing all EDUs starting with a discourse cue (“marked”) and one containing the remaining EDUs (“unmarked”). We hypothesize that the effect of the discourse-aware model should be stronger on the first subset, since the presence of the connective indicates the possibility of a discourse relation with the previous EDU. The set of discourse connectives is taken from the Penn Discourse Treebank (Prasad et al., 2008), thus creating a list of 240 potential connectives.

Table 5 presents a subset of “marked” EDUs for which trying to assign the sentiment and aspect out of context (i.e. without the previous EDU) is a difficult task. In examples 1-3 there is no explicit mention of the aspect. However, there is an anaphoric expression (marked in bold) which

refers to a mention of the aspect in some previous EDU. On the other hand, in examples 4 and 5 there is an ambiguity in the choice of aspect; in example 5, *tea making facilities* can refer to a breakfast at the hotel (label *food*) or to facilities in the room (label *rooms*). Finally, examples 6-8 are too short and not informative at all which indicates that the segmentation tool does not always predict a desirable segmentation. Consequently, automatic induction of segmentation may be a better option.

Table 4 presents quantitative results of this analysis. Although the performance over the “unmarked” example is the same for the two models, this is not the case for the “marked” instances where the discourse-informed model leverages the discourse signal and achieves better performance. This behavior agrees with our initial hypothesis, and suggests that our discourse representation, though application-specific, relies in part on the information encoded in linguistically-defined discourse cues. We will confirm this intuition in the qualitative evaluation section. The increase for the “marked” EDUs does not translate into greater differences for the overall scores (Table 3) as marked relations are considerably less frequent than unmarked ones in our gold standard (i.e. 35% of the EDUs are “marked”). Nevertheless, this clearly suggests that the discourse-informed model is in fact capable of exploiting discourse signal.

## 5.2 Qualitative analysis

To investigate the quality of the induced discourse structure, we present the most frequent discourse cues extracted for every discourse relation. Table 6 presents a selection of cues that best explain the discourse relation they have been associated with. A general observation is that among the cues there are not only “traditional” discourse connectives like *even though*, *although*, *and*, but also cues that are discriminative for the specific application.

In relation *SameAlt* we can mostly observe phrases that tend to introduce a new aspect, since an explicit mention of it is provided (e.g. *the location is*, *the room was*) and more specific phrases like *in addition* are used to introduce a new aspect with the same sentiment. However, these cues reveal important information about the aspect of the EDU, and since they are associated with the language model  $\tilde{\phi}$ , they are not visible anymore to the language model of aspects  $\phi$ .

Cues for the relation *AltSame* also include

Discourse relation	Discourse Cues
SameAlt	<b>the location is</b> , <b>the room was</b> , the hotel has, <b>and the room</b> , and the bed, breakfast was, the staff were, <b>in addition</b> , good luck
AltSame	<b>but</b> , and, it was, and it was, and they, although, and it, but it, <b>but it was</b> , however, which was, this is, <b>this was</b> , they were, the only thing, even though, <b>unfortunately</b> , needless to say, fortunately
AltAlt	<b>the room was</b> , the staff were, <b>the only</b> , the hotel is, but the, however, also, or, <b>overall i</b> , unfortunately, <b>we will definitely</b> , on the plus, the only downside , even though, and even though, <b>i would definately</b>

Table 6: Induced cues from the discourse relations

phrases that contain some anaphoric expressions, which might refer to previous mentions of an aspect in the discourse (i.e. previous EDU). We expect that since there is an anaphoric expression, explicit lexical features for the aspect will be missing, making thus the decision concerning aspect assignment ambiguous for any discourse-agnostic model. Interestingly, we found the expressions *unfortunately*, *fortunately*, *the only thing* in the same relation, since all indicate a change in sentiment. Finally, *AltAlt* can be viewed as a mixture of the other two relations. Furthermore, for this relation we can find expressions that tend to be used at the end of a review, since at this point we normally change the aspect and often even sentiment. Some examples of these cases are *overall*, *we will definitely* and even the misspelled version of the latter *i would definately*.

## 5.3 Features in supervised learning

As an additional experiment to demonstrate informative of the output of the two models, we design a supervised learning task of predicting sentiment and topic of EDUs. In this setting, the feature vector of every EDU consists of its bag-of-word-representation to which we add two extra features; the models’ predictions of topic and sentiment. We train a support vector machine with a polynomial kernel using the default parameters of Weka<sup>8</sup> and perform 10-fold cross-validation.

Table 7 presents results of this analysis in terms of accuracy for four classification tasks, i.e. predicting both sentiment and topic, only sentiment and only topic for all EDUs, as well as predicting sentiment and topic for the “marked” dataset. First, we observe that incorporation of the topic-

<sup>8</sup><http://www.cs.waikato.ac.nz/ml/weka/>

Features	aspect+sentiment (28 classes)	aspect (10 classes)	sentiment (3 classes)	Marked only sentiment+aspect (28 classes)
<i>only unigrams</i>	36.3	49.8	57.1	26.2
<i>unigrams + SentAsp</i>	38.0	50.4	59.3	27.8
<i>unigrams + Discourse</i>	<b>39.1</b>	<b>52.4</b>	<b>59.4</b>	<b>29.1</b>

Table 7: Supervised learning at the EDU level (accuracy)

model features on a *unigram-only* model results in an improvement in classification performance across all tasks (predicting sentiment, predicting aspects, or both); as a matter of fact, our accuracy results for predicting sentiment are comparable to the sentence-level results presented by Täckström and McDonald (2011). We have to stress that accuracies for the joint task (i.e. predicting both sentiment and topic) are expected to be lower since it can also be seen as the product of the two other tasks (i.e. predicting only sentiment and only topic). We also observe that the features induced from the *Discourse* model result in higher accuracy than the ones from the discourse-agnostic model *SentAsp* both in the complete set of EDUs and the “marked” subset, results that are in line with the ones presented in Table 4. Finally, the fact that the results for the complete set of EDUs are higher than the ones for the “marked” dataset clearly suggests that the latter constitute a hard case for sentiment analysis, in which exploiting discourse signal proves to be beneficial.

## 6 Related Work

Recently, there has been significant interest in leveraging content structure for a number of NLP tasks (Webber et al., 2011). Sentiment analysis has not been an exception to this and discourse has been used in order to enforce constraints on the assignment of polarity labels at several granularity levels, ranging from the lexical level (Polanyi and Zaenen, 2006) to the review level (Taboada et al., 2011). One way to deal with this problem is to model the interactions by using a pre-compiled set of polarity shifters (Nakagawa et al., 2010; Polanyi and Zaenen, 2006; Sadamitsu et al., 2008). Socher et al. (2011) defined a recurrent neural network model, which, in essence, learns those polarity shifters relying on sentence-level sentiment labels. Though successful, this model is unlikely to capture intra-sentence non-local phenomena such as effect of discourse connectives, unless it is provided with syntactic information as an input. This may be problematic for the noisy sentiment-analysis domain and especially

for poor-resource languages. Similar to our work, others have focused on modeling interactions between phrases and sentences. However, this has been achieved by either using a subset of relations that can be found in discourse theories (Zhou et al., 2011; Asher et al., 2008; Snyder and Barzilay, 2007) or by using directly (Taboada et al., 2008) the output of discourse parsers (Soricut and Marcu, 2003). Discourse cues as predictive features of topic boundaries have also been considered in Eisenstein and Barzilay (2008). This work was extended by Trivedi and Eisenstein (2013), where discourse connectors are used as features for modeling subjectivity transitions.

Another related line of research was presented in Somasundaran et al. (2009) where a domain-specific discourse scheme is considered. Similarly to our set-up, discourse relations enforce constraints on sentiment polarity of associated sentiment expressions. Somasundaran et al. (2009) show that gold-standard discourse information encoded in this way provides a useful signal for prediction of sentiment, but they leave automatic discourse relation prediction for future work. They use an integer linear programming framework to enforce agreement between classifiers and soft constraints provided by discourse annotations. This contrasts with our work; we do not rely on expert discourse annotation, but rather induce both discourse relations and cues jointly with aspect and sentiment.

## 7 Conclusions and Future Work

In this work, we showed that by jointly inducing discourse information in the form of discourse cues, we can achieve better predictions for aspect-specific sentiment polarity. Our contribution consists in proposing a general way of how discourse information can be integrated in any LDA-style discourse-agnostic model of aspect and sentiment. In the future, we aim at modeling more flexible sets of discourse relations and automatically inducing discourse segmentation relevant to the task.

## References

- Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In *Proceedings of the SemEval*, pages 7–12.
- Nicholas Asher, Farah Benamara, and Yvette Yannick Mathieu. 2008. Distilling opinion in discourse: A preliminary study. *Proceedings of Coling*, pages 5–8.
- Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Proceedings of NAACL*, pages 804–812.
- Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of EMNLP*, pages 334–343.
- Emily B Fox, Erik B Sudderth, Michael I Jordan, and Alan S Willsky. 2008. An HDP-HMM for systems with state persistence. In *Proceedings of ICML*.
- Gayatree Ganu, Noemie Elhadad, and Amelie Marian. 2009. Beyond the stars: Improving rating predictions using review text content. In *Proceedings of WebDB*.
- Sharon Goldwater, Thomas L Griffiths, and Mark Johnson. 2009. A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.
- Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235.
- Geoffrey E Hinton. 1999. Products of experts. In *Proceedings of ICANN*, volume 1, pages 1–6.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of ACM SIGKDD*, pages 168–177.
- Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. 2012. Incorporating lexical priors into topic models. *Proceedings of EACL*, pages 204–213.
- Yohan Jo and Alice H Oh. 2011. Aspect and sentiment unification model for online review analysis. In *Proceedings of WSDM*, pages 815–824.
- Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceeding of CIKM*, pages 375–384.
- William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.
- Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of WWW*, pages 171–180.
- Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using crfs with hidden variables. In *Proceedings of NAACL*, pages 786–794.
- Tahira Naseem, Benjamin Snyder, Jacob Eisenstein, and Regina Barzilay. 2009. Multilingual part-of-speech tagging: Two unsupervised approaches. *Journal of Artificial Intelligence Research*, 36(1):341–385.
- Livia Polanyi and Annie Zaenen. 2006. Contextual valence shifters. *Computing attitude and affect in text: Theory and applications*, pages 1–10.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltasakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *Proceedings of LREC*.
- Kugatsu Sadamitsu, Satoshi Sekine, and Mikio Yamamoto. 2008. Sentiment analysis based on probabilistic models using inter-sentence information. In *Proceedings of ACL*, pages 2892–2896.
- Benjamin Snyder and Regina Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *Proceedings of HLT-NAACL*, pages 300–307.
- Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of EMNLP*, pages 151–161.
- Swapna Somasundaran, Janyce Wiebe, and Josef Ruppenhofer. 2008. Discourse level opinion interpretation. In *Proceedings of Coling*, pages 801–808.
- Swapna Somasundaran, Galileo Namata, Janyce Wiebe, and Lise Getoor. 2009. Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of EMNLP*, pages 170–179.
- Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of NAACL*, pages 149–156.
- Maite Taboada, Kimberly Voll, and Julian Brooke. 2008. Extracting sentiment as a function of discourse structure and topicality. *Simon Fraser University, Tech. Rep.*, 20.
- Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307.
- Oscar Täckström and Ryan McDonald. 2011. Semi-supervised latent variable models for sentence-level sentiment analysis. In *Proceedings of ACL*, pages 569–574.
- Ivan Titov and Ryan McDonald. 2008a. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL*, pages 308–316.

- Ivan Titov and Ryan McDonald. 2008b. Modeling on-line reviews with multi-grain topic models. In *Proceedings of WWW*, pages 112–120.
- Rakshit Trivedi and Jacob Eisenstein. 2013. Discourse connectors for latent subjectivity in sentiment analysis. In *Proceedings of NAACL*.
- Peter D Turney and Michael L Littman. 2002. Unsupervised learning of semantic orientation from a hundred-billion-word corpus.
- Kimberly Voll and Maite Taboada. 2007. Not all words are created equal: Extracting semantic orientation as a function of adjective relevance. In *Proceedings of Australian Conf. on AI*.
- Bonnie Webber, Markus Egg, and Valia Kordoni. 2011. Discourse structure and language technology. *Natural Language Engineering*, 1(1):1–54.
- Lanjun Zhou, Binyang Li, Wei Gao, Zhongyu Wei, and Kam-Fai Wong. 2011. Unsupervised discovery of discourse relations for eliminating intra-sentence polarity ambiguities. In *Proceedings EMNLP*, pages 162–171.

# Joint Inference for Fine-grained Opinion Extraction

**Bishan Yang**

Department of Computer Science  
Cornell University  
bishan@cs.cornell.edu

**Claire Cardie**

Department of Computer Science  
Cornell University  
cardie@cs.cornell.edu

## Abstract

This paper addresses the task of fine-grained opinion extraction – the identification of opinion-related entities: the opinion expressions, the opinion holders, and the targets of the opinions, and the relations between opinion expressions and their targets and holders. Most existing approaches tackle the extraction of opinion entities and opinion relations in a pipelined manner, where the interdependencies among different extraction stages are not captured. We propose a joint inference model that leverages knowledge from predictors that optimize subtasks of opinion extraction, and seeks a globally optimal solution. Experimental results demonstrate that our joint inference approach significantly outperforms traditional pipeline methods and baselines that tackle subtasks in isolation for the problem of opinion extraction.

## 1 Introduction

Fine-grained opinion analysis is concerned with identifying opinions in text at the expression level; this includes identifying the subjective (i.e., opinion) expression itself, the opinion holder and the target of the opinion (Wiebe et al., 2005). The task has received increasing attention as many natural language processing applications would benefit from the ability to identify text spans that correspond to these key components of opinions. In question-answering systems, for example, users may submit questions in the form “What does entity A think about target B?”; opinion-oriented summarization systems also need to recognize opinions and their targets and holders.

In this paper, we address the task of identifying opinion-related entities and opinion relations. We

consider three types of opinion entities: *opinion expressions* or direct subjective expressions as defined in Wiebe et al. (2005) — expressions that explicitly indicate emotions, sentiment, opinions or other private states (Quirk et al., 1985) or speech events expressing private states; *opinion targets* — expressions that indicate what the opinion is about; and *opinion holders* — mentions of whom or what the opinion is from. Consider the following examples in which opinion expressions (O) are underlined and targets (T) and holders (H) of the opinion are bracketed.

S1: [The workers]<sub>[H<sub>1,2</sub>]</sub> were irked<sub>[O<sub>1</sub>]</sub> by [the government report]<sub>[T<sub>1</sub>]</sub> and were worried<sub>[O<sub>2</sub>]</sub> as they went about their daily chores.

S2: From the very start it could be predicted<sub>[O<sub>1</sub>]</sub> that on the subject of economic globalization, [the developed states]<sub>[T<sub>1,2</sub>]</sub> were going to come across fierce opposition<sub>[O<sub>2</sub>]</sub>.

The numeric subscripts denote *linking relations*, one of IS-ABOUT or IS-FROM. In S1, for instance, opinion expression “were irked” ( $O_1$ ) IS-ABOUT “the government report” ( $T_1$ ). Note that the IS-ABOUT relation can contain an empty target (e.g. “were worried” in S1); similarly for IS-FROM w.r.t. the opinion holder (e.g. “predicted” in S2). We also allow an opinion entity to be involved in multiple relations (e.g. “the developed states” in S2).

Not surprisingly, fine-grained opinion extraction is a challenging task due to the complexity and variety of the language used to express opinions and their components (Pang and Lee, 2008). Nevertheless, much progress has been made in extracting opinion information from text. Sequence labeling models have been successfully employed to identify opinion expressions (e.g. (Breck et al.,

2007; Yang and Cardie, 2012)) and relation extraction techniques have been proposed to extract opinion holders and targets based on their linking relations to the opinion expressions (e.g. Kim and Hovy (2006), Kobayashi et al. (2007)). However, most existing work treats the extraction of different opinion entities and opinion relations in a pipelined manner: the interaction between different extraction tasks is not modeled jointly and error propagation is not considered. One exception is Choi et al. (2006), which proposed an ILP approach to jointly identify opinion holders, opinion expressions and their IS-FROM linking relations, and demonstrated the effectiveness of joint inference. Their ILP formulation, however, does not handle *implicit linking relations*, i.e. opinion expressions with no explicit opinion holder; nor does it consider IS-ABOUT relations.

In this paper, we present a model that jointly identifies opinion-related entities, including *opinion expressions*, *opinion targets* and *opinion holders* as well as the associated opinion linking relations, IS-ABOUT and IS-FROM. For each type of opinion relation, we allow implicit (i.e. empty) arguments for cases when the opinion holder or target is not explicitly expressed in text. We model entity identification as a sequence tagging problem and relation extraction as binary classification. A joint inference framework is proposed to jointly optimize the predictors for different sub-problems with constraints that enforce global consistency. We hypothesize that the ambiguity in the extraction results will be reduced and thus, performance increased. For example, uncertainty w.r.t. the spans of opinion entities can adversely affect the prediction of opinion relations; and evidence of opinion relations might provide clues to guide the accurate extraction of opinion entities.

We evaluate our approach using a standard corpus for fine-grained opinion analysis (the MPQA corpus (Wiebe et al., 2005)) and demonstrate that our model outperforms by a significant margin traditional baselines that do not employ joint inference for extracting opinion entities and different types of opinion relations.

## 2 Related Work

Significant research effort has been invested into fine-grained opinion extraction for open-domain text such as news articles (Wiebe et al., 2005; Wilson et al., 2009). Many techniques were proposed

to identify the text spans for opinion expressions (e.g. (Breck et al., 2007; Johansson and Moschitti, 2010b; Yang and Cardie, 2012)), opinion holders (e.g. (Choi et al., 2005)) and topics of opinions (Stoyanov and Cardie, 2008). Some consider extracting opinion targets/holders along with their relation to the opinion expressions. Kim and Hovy (2006) identifies opinion holders and targets by using their semantic roles related to opinion words. Ruppenhofer et al. (2008) argued that semantic role labeling is not sufficient for identifying opinion holders and targets. Johansson and Moschitti (2010a) extract opinion expressions and holders by applying reranking on top of sequence labeling methods. Kobayashi et al. (2007) considered extracting “aspect-evaluation” relations (relations between opinion expressions and targets) by identifying opinion expressions first and then searching for the most likely target for each opinion expression via a binary relation classifier. All these methods extract opinion arguments and opinion relations in separate stages instead of extracting them jointly.

Most similar to our method is Choi et al. (2006), which jointly extracts opinion expressions, holders and their IS-FROM relations using an ILP approach. In contrast, our approach (1) also considers the IS-ABOUT relation which is arguably more complex due to the larger variety in the syntactic structure exhibited by opinion expressions and their targets, (2) handles implicit opinion relations (opinion expressions without any associated argument), and (3) uses a simpler ILP formulation.

There has also been substantial interest in opinion extraction from product reviews (Liu, 2012). Most existing approaches focus on the extraction of opinion targets and their associated opinion expressions and usually employ a pipeline architecture: generate candidates of opinion expressions and opinion targets first, and then use rule-based or machine-learning-based approaches to identify potential relations between opinions and targets (Hu and Liu, 2004; Wu et al., 2009; Liu et al., 2012). In addition to pipeline approaches, bootstrapping-based approaches were proposed (Qiu et al., 2009; Qiu et al., 2011; Zhang et al., 2010) to identify opinion expressions and targets iteratively; however, they suffer from the problem of error propagation.

There is much work demonstrating the benefit of performing global inference. Roth and Yih

(2004) proposed a global inference approach in the formulation of a linear program (LP) and applied it to the task of extracting named entities and relations simultaneously. Their problem is similar to ours — the difference is that Roth and Yih Roth and Yih (2004) assume that named entity spans are known a priori and only their labels need to be assigned. Joint inference has also been applied to semantic role labeling (Punyakanok et al., 2008; Srikumar and Roth, 2011; Das et al., 2012), where the goal is to jointly identify semantic arguments for given lexical predicates. The problem is conceptually similar to identifying opinion arguments for opinion expressions, however, we do not assume prior knowledge of opinion expressions (unlike in SRL, where predicates are given).

### 3 Model

As proposed in Section 1, we consider the task of jointly identifying opinion entities and opinion relations. Specifically, given a sentence, our goal is to identify spans of opinion expressions, opinion arguments (targets and holders) and their associated linking relations. Training data consists of text with manually annotated opinion expression and argument spans, each with a list of relation ids specifying the linking relation between opinion expressions and their arguments.

In this section, we will describe how we model opinion entity identification and opinion relation extraction, and how we combine them in a joint inference model.

#### 3.1 Opinion Entity Identification

We formulate the task of opinion entity identification as a sequence labeling problem and employ conditional random fields (CRFs) (Lafferty et al., 2001) to learn the probability of a sequence assignment  $\mathbf{y}$  for a given sentence  $\mathbf{x}$ . Through inference we can find the best sequence assignment for sentence  $x$  and recover the opinion entities according to the standard “IOB” encoding scheme. We consider four entity labels:  $D, T, H, N$ , where  $D$  denotes opinion expressions,  $T$  denotes opinion targets,  $H$  denotes opinion holders and  $N$  denotes “NONE” entities.

We define potential function  $f_{iz}$  that gives the probability of assigning a span  $i$  with entity label  $z$ , and the probability is estimated based on the learned parameters from CRFs. Formally, given a within-sentence span  $i = (a, b)$ , where  $a$  is the

starting position and  $b$  is the end position, and label  $z \in \{D, T, H\}$ , we have

$$f_{iz} = p(\mathbf{y}_a = B_z, \mathbf{y}_{a+1} = I_z, \dots, \mathbf{y}_b = I_z, \mathbf{y}_{b+1} \neq I_z | \mathbf{x})$$

$$f_{iN} = p(\mathbf{y}_a = O, \dots, \mathbf{y}_b = O | \mathbf{x})$$

These probabilities can be efficiently computed using the forward-backward algorithm.

#### 3.2 Opinion Relation Extraction

We consider extracting the IS-ABOUT and IS-FROM opinion relations. In the following we will not distinguish these two relations, since they can both be characterized as relations between opinion expressions and opinion arguments, and the methods for relation extraction are the same.

We treat the relation extraction problem as a combination of two binary classification problems: *opinion-arg classification*, which decides whether a pair consisting of an opinion candidate  $o$  and an argument candidate  $a$  forms a relation; and *opinion-implicit-arg classification*, which decides whether an opinion candidate  $o$  is linked to an implicit argument, i.e. no argument is mentioned. We define a potential function  $r$  to capture the strength of association between an opinion candidate  $o$  and an argument candidate  $a$ ,

$$r_{oa} = p(y = 1 | x) - p(y = 0 | x)$$

where  $p(y = 1 | x)$  and  $p(y = 0 | x)$  are the logistic regression estimates of the positive and negative relations. Similarly, we define potential  $r_{o\emptyset}$  to denote the confidence of predicting opinion span  $o$  associated with an implicit argument.

##### 3.2.1 Opinion-Arg Relations

For *opinion-arg* classification, we construct candidates of opinion expressions and opinion arguments and consider each pair of an opinion candidate and an argument candidate as a potential opinion relation. Conceptually, all possible subsequences in the sentence are candidates. To filter out candidates that are less reasonable, we consider the opinion expressions and arguments obtained from the  $n$ -best predictions by CRFs<sup>1</sup>. We also employ syntactic patterns from dependency

<sup>1</sup>We randomly split the training data into 10 parts and obtained the 50-best CRF predictions on each part for the generation of candidates. We also experimented with candidates generated from more CRF predictions, but did not find any performance improvement for the task.

trees to generate candidates. Specifically, we selected the most common patterns of the shortest dependency paths<sup>2</sup> between an opinion candidate  $o$  and an argument candidate  $a$  in our dataset, and include all pairs of candidates that satisfy at least one dependency pattern. For the IS-ABOUT relation, the top three patterns are (1)  $o \uparrow_{dobj} a$ , (2)  $o \uparrow_{ccomp} x \uparrow_{nsubj} a$  ( $x$  is a word in the path that is not covered by either  $o$  nor  $a$ ), (3)  $o \uparrow_{ccomp} a$ ; for the IS-FROM relation, the top three patterns are (1)  $o \uparrow_{nsubj} a$ , (2)  $o \uparrow_{poss} a$ , (3)  $o \downarrow_{ccomp} x \uparrow_{nsubj} a$ .

Note that generating candidates this way will give us a large number of negative examples. Similar to the preprocessing approach in (Choi et al., 2006), we filter pairs of opinion and argument candidates that do not overlap with any gold standard relation in our training data.

Many features we use are common features in the SRL tasks (Punyakanok et al., 2008) due to the similarity of opinion relations to the predicate-argument relations in SRL (Ruppenhofer et al., 2008; Choi et al., 2006). In general, the features aim to capture (a) local properties of the candidate opinion expressions and arguments and (b) syntactic and semantic attributes of their relation.

**Words and POS tags:** the words contained in the candidate and their POS tags.

**Lexicon:** For each word in the candidate, we include its WordNet hypernyms and its strength of subjectivity in the Subjectivity Lexicon<sup>3</sup> (e.g. weaksubj, strongsubj).

**Phrase type:** the syntactic category of the deepest constituent that covers the candidate in the parse tree, e.g. NP, VP.

**Semantic frames:** For each verb in the opinion candidate, we include its frame types according to FrameNet<sup>4</sup>.

**Distance:** the relative distance (number of words) between the opinion and argument candidates.

**Dependency Path:** the shortest path in the dependency tree between the opinion candidate and the target candidate, e.g.  $ccomp \uparrow nsubj \uparrow$ . We also include word types and POS types in the paths, e.g.  $opinion \uparrow_{ccomp} suffering \uparrow_{nsubj} patient$ ,

<sup>2</sup>We use the Stanford Parser to generate parse trees and dependency graphs.

<sup>3</sup>[http://mpqa.cs.pitt.edu/lexicons/subj\\_lexicon/](http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/)

<sup>4</sup><https://framenet.icsi.berkeley.edu/fndrupal/>

$NN \uparrow_{ccomp} VBG \uparrow_{nsubj} NN$ . The dependency path has been shown to be very useful in extracting opinion expressions and opinion holders (Johansson and Moschitti, 2010a).

### 3.2.2 Opinion-Implicit-Arg Relations

When the *opinion-arg* relation classifier predicts that there is no suitable argument for the opinion expression candidate, it does not capture the possibility that an opinion candidate may associate with an implicit argument. To incorporate knowledge of implicit relations, we build an *opinion-implicit-arg* classifier to identify an opinion candidate with an implicit argument based on its own properties and context information.

For training, we consider all gold-standard opinion expressions as training examples — including those with implicit arguments — as positive examples and those associated with explicit arguments as negative examples. For features, we use words, POS tags, phrase types, lexicon and semantic frames (see Section 3.2.1 for details) to capture the properties of the opinion expression, and also features that capture the context of the opinion expression:

**Neighboring constituents:** The words and grammatical roles of neighboring constituents of the opinion expression in the parse tree — the left and right sibling of the deepest constituent containing the opinion expression in the parse tree.

**Parent Constituent:** The grammatical role of the parent constituent of the deepest constituent containing the opinion expression.

**Dependency Argument:** The word types and POS types of the arguments of the dependency patterns in which the opinion expression is involved. We consider the same dependency patterns that are used to generate candidates for *opinion-arg* classification.

### 3.3 Joint Inference

The inference goal is to find the optimal prediction for both opinion entity identification and opinion relation extraction. For a given sentence, we denote  $\mathcal{O}$  as a set of opinion candidates,  $\mathcal{A}_k$  as a set of argument candidates, where  $k$  denotes the type of opinion relation — IS-ABOUT or IS-FROM — and  $\mathcal{S}$  as a set of within-sentence spans that cover all of the opinion candidates and argument can-

didates. We introduce binary variable  $x_{iz}$ , where  $x_{iz} = 1$  means span  $i$  is associated with label  $z$ . We also introduce binary variable  $u_{ij}$  for every pair of opinion candidate  $i$  and argument candidate  $j$ , where  $u_{ij} = 1$  means  $i$  forms an opinion relation with  $j$ , and binary variable  $v_{ik}$  for every opinion candidate  $i$  in relation type  $k$ , where  $v_{ik} = 1$  means  $i$  associates with an implicit argument in relation  $k$ . Given the binary variables  $x_{iz}, u_{ij}, v_{ik}$ , it is easy to recover the entity and relation assignment by checking which spans are labeled as opinion entities, and which opinion span and argument span form an opinion relation.

The objective function is defined as a linear combination of the potentials from different predictors with a parameter  $\lambda$  to balance the contribution of two components: opinion entity identification and opinion relation extraction.

$$\begin{aligned} & \arg \max_{x,u,v} \lambda \sum_{i \in \mathcal{S}} \sum_z f_{iz} x_{iz} \\ & + (1 - \lambda) \sum_k \sum_{i \in \mathcal{O}} \left( \sum_{j \in \mathcal{A}_k} r_{ij} u_{ij} + r_{i\emptyset} v_{ik} \right) \end{aligned} \quad (1)$$

It is subject to the following linear constraints:

Constraint 1: *Uniqueness*. For each span  $i$ , we must assign one and only one label  $z$ , where  $z \in \{H, D, T, N\}$ .

$$\sum_z x_{iz} = 1$$

Constraint 2: *Non-overlapping*. If two spans  $i$  and  $j$  overlap, then at most one of the spans can be assigned to a non-NONE entity label:  $H, D, T$ .

$$\sum_{z \neq N} x_{iz} + \sum_{z \neq N} x_{jz} \leq 1$$

Constraint 3: *Consistency between the opinion-arg and opinion-implicit-arg classifiers*. For an opinion candidate  $i$ , if it is predicted to have an implicit argument in relation  $k$ ,  $v_{ik} = 1$ , then no argument candidate should form a relation with  $i$ . If  $v_{ik} = 0$ , then there exists some argument candidate  $j \in \mathcal{A}_k$  such that  $u_{ij} = 1$ . We introduce two auxiliary binary variables  $a_{ik}$  and  $b_{ik}$  to limit the maximum number of relations associated with each opinion candidate to be less than or equal to

three<sup>5</sup>. When  $v_{ik} = 1$ ,  $a_{ik}$  and  $b_{ik}$  have to be 0.

$$\sum_{j \in \mathcal{A}_k} u_{ij} = 1 - v_{ik} + a_{ik} + b_{ik}$$

$$a_{ik} \leq 1 - v_{ik}, b_{ik} \leq 1 - v_{ik}$$

Constraint 4: *Consistency between opinion-arg classifier and opinion entity extractor*. Suppose an argument candidate  $j$  in relation  $k$  is assigned an argument label by the entity extractor, that is  $x_{jz} = 1$  ( $z = T$  for IS-ABOUT relation and  $z = H$  for IS-FROM relation), then there exists some opinion candidates that associate with  $j$ . Similar to constraint 3, we introduce auxiliary binary variables  $c_j$  and  $d_j$  to enforce that an argument  $j$  links to at most three opinion expressions. If  $x_{jz} = 0$ , then no relations should be extracted for  $j$ .

$$\sum_{i \in \mathcal{O}} u_{ij} = x_{jz} + c_{jk} + d_{jk}$$

$$c_{jk} \leq x_{jz}, d_{jk} \leq x_{jz}$$

Constraint 5: *Consistency between the opinion-implicit-arg classifier and opinion entity extractor*. When an opinion candidate  $i$  is predicted to associate with an implicit argument in relation  $k$ , that is  $v_{ik} = 1$ , then we allow  $x_{iD}$  to be either 1 or 0 depending on the confidence of labeling  $i$  as an opinion expression. When  $v_{ik} = 0$ , there exists some opinion argument associated with the opinion candidate, and we enforce  $x_{iD} = 1$ , which means the entity extractor agrees to label  $i$  as an opinion expression.

$$v_{ik} + x_{iD} \geq 1$$

Note that in our ILP formulation, the label assignment for a candidate span involves one multiple-choice decision among different opinion entity labels and the ‘‘NONE’’ entity label. The scores of different label assignments are comparable for the same span since they come from one entity extraction model. This makes our ILP formulation advantageous over the ILP formulation proposed in Choi et al. (2006), which needs  $m$  binary decisions for a candidate span, where  $m$  is the number of types of opinion entities, and the score for each possible label assignment is obtained by

<sup>5</sup>It is possible to add more auxiliary variables to allow more than three arguments to link to an opinion expression, but this rarely happens in our experiments. For the IS-FROM relation, we set  $a_{ik} = 0, b_{ik} = 0$  since an opinion expression usually has only one holder.

the sum of raw scores from  $m$  independent extraction models. This design choice also allows us to easily deal with multiple types of opinion arguments and opinion relations.

## 4 Experiments

For evaluation, we used version 2.0 of the MPQA corpus (Wiebe et al., 2005; Wilson, 2008), a widely used data set for fine-grained opinion analysis.<sup>6</sup> We considered the subset of 482 documents<sup>7</sup> that contain attitude and target annotations. There are a total of 9,471 sentences with opinion-related labels at the phrase level. We set aside 132 documents as a development set and use 350 documents as the evaluation set. All experiments employ 10-fold cross validation on the evaluation set; the average over the 10 runs is reported.

Our gold standard opinion expressions, opinion targets and opinion holders correspond to the direct subjective annotations, target annotations and agent annotations, respectively. The IS-FROM relation is obtained from the *agent* attribute of each opinion expression. The IS-ABOUT relation is obtained from the attitude annotations: each opinion expression is annotated with attitude frames and each attitude frame is associated with a list of targets. The relations may overlap: for example, in the following sentence, the target of relation 1 contains relation 2.

[John]<sub>H<sub>1</sub></sub> is happy<sub>O<sub>1</sub></sub> because [[he]<sub>H<sub>2</sub></sub>  
loves<sub>O<sub>2</sub></sub> [being at Enderly Park]<sub>T<sub>2</sub></sub>]<sub>T<sub>1</sub></sub>.

We discard relations that contain sub-relations because we believe that identifying the sub-relations usually is sufficient to recover the discarded relations. (Prediction of overlapping relations is considered as future work.) In the example above, we will identify (*loves, being at Enderly Park*) as an IS-ABOUT relation and *happy* as an opinion expression associated with an implicit target. Table 1 shows some statistics of the corpus.

We adopted the evaluation metrics for entity and relation extraction from Choi et al. (2006), which include precision, recall, and F1-measure according to *overlap* and *exact* matching metrics.<sup>8</sup> We

<sup>6</sup>Available at <http://www.cs.pitt.edu/mpqa/>.

<sup>7</sup>349 news articles from the original MPQA corpus, 84 Wall Street Journal articles (Xbank), and 48 articles from the American National Corpus.

<sup>8</sup>Overlap matching considers two spans to match if they overlap, while exact matching requires two spans to be exactly the same.

	Opinion	Target	Holder
TotalNum	5849	4676	4244
	Opinion-arg Relations		Implicit Relations
IS-ABOUT	4823		1302
IS-FROM	4662		1187

Table 1: Data Statistics of the MPQA Corpus.

will focus our discussion on results obtained using overlap matching, since the exact boundaries of opinion entities are hard to define even for human annotators (Wiebe et al., 2005).

We trained CRFs for opinion entity identification using the following features: indicators for words, POS tags, and lexicon features (the subjectivity strength of the word in the Subjectivity Lexicon). All features are computed for the current token and tokens in a  $[-1, +1]$  window. We used L2-regularization; the regularization parameter was tuned using the development set. We trained the classifiers for relation extraction using L1-regularized logistic regression with default parameters using the LIBLINEAR (Fan et al., 2008) package. For joint inference, we used GLPK<sup>9</sup> to provide the optimal ILP solution. The parameter  $\lambda$  was tuned using the development set.

### 4.1 Baseline Methods

We compare our approach to several pipeline baselines. Each extracts opinion entities first using the same CRF employed in our approach, and then predicts opinion relations on the opinion entity candidates obtained from the CRF prediction. Three relation extraction techniques were used in the baselines:

- Adj: Inspired by the adjacency rule used in Hu and Liu (2004), it links each argument candidate to its nearest opinion candidate. Arguments that do not link to any opinion candidate are discarded. This is also used as a strong baseline in Choi et al. (2006).
- Syn: Links pairs of opinion and argument candidates that present prominent syntactic patterns. (We consider the syntactic patterns listed in Section 3.2.1.) Previous work also demonstrates the effectiveness of syntactic information in opinion extraction (Johansson and Moschitti, 2012).

<sup>9</sup><http://www.gnu.org/software/glpk/>

Method	Opinion Expression			Opinion Target			Opinion Holder		
	P	R	F1	P	R	F1	P	R	F1
CRF	82.21	66.15	73.31	73.22	48.58	58.41	72.32	49.09	58.48
CRF+Adj	82.21	66.15	73.31	80.87	42.31	55.56	75.24	48.48	58.97
CRF+Syn	82.21	66.15	73.31	81.87	30.36	44.29	78.97	40.20	53.28
CRF+RE	83.02	48.99	61.62	85.07	22.01	34.97	78.13	40.40	53.26
Joint-Model	71.16	77.85	<b>74.35*</b>	75.18	57.12	<b>64.92**</b>	67.01	66.46	<b>66.73**</b>
CRF	66.60	52.57	58.76	44.44	29.60	35.54	65.18	44.24	52.71
CRF+Adj	66.60	52.57	58.76	49.10	25.81	33.83	68.03	43.84	53.32
CRF+Syn	66.60	52.57	58.76	50.26	18.41	26.94	74.60	37.98	50.33
CRF+RE	69.27	40.09	50.79	60.45	15.37	24.51	75	38.79	51.13
Joint-Model	57.39	62.40	<b>59.79*</b>	49.15	38.33	<b>43.07**</b>	62.73	62.22	<b>62.47**</b>

Table 2: Performance on opinion entity extraction using *overlap* and *exact* matching metrics (the top table uses *overlap* and the bottom table uses *exact*). Two-tailed t-test results are shown on F1 measure for our method compared to the other baselines (statistical significance is indicated with \* ( $p < 0.05$ ), \*\* ( $p < 0.005$ )).

Method	IS-ABOUT			IS-FROM		
	P	R	F1	P	R	F1
CRF+Adj	73.65	37.34	49.55	70.22	41.58	52.23
CRF+Syn	76.21	28.28	41.25	77.48	36.63	49.74
CRF+RE	78.26	20.33	32.28	74.81	37.55	50.00
CRF+Adj-merged-10-best	25.05	61.18	35.55	30.28	62.82	40.87
CRF+Syn-merged-10-best	41.60	45.66	43.53	48.08	54.03	50.88
CRF+RE-merged-10-best	51.60	33.09	40.32	47.73	54.40	50.84
Joint-Model	64.38	51.20	<b>57.04**</b>	64.97	58.61	<b>61.63**</b>

Table 3: Performance on opinion relation extraction using the *overlap* metric.

- RE: Predicts opinion relations by employing the *opinion-arg* classifier and *opinion-implicit-arg* classifier. First, the *opinion-arg* classifier identifies pairs of opinion and argument candidates that form valid opinion relations, and then the *opinion-implicit-arg* classifier is used on the remaining opinion candidates to further identify opinion expressions without explicit arguments.

We report results using opinion entity candidates from the best CRF output and from the merged 10-best CRF output.<sup>10</sup> The motivation of merging the 10-best output is to increase recall for the pipeline methods.

## 5 Results

Table 2 shows the results of opinion entity identification using both *overlap* and *exact* metrics. We compare our approach with the pipeline baselines and CRF (the first step of the pipeline). We can see that our joint inference approach significantly outperforms all the baselines in F1 measure on extracting all types of opinion entities. In general,

<sup>10</sup>It is similar to the *merged 10-best* baseline in Choi et al. (2006). If an entity  $E_i$  extracted by the  $i$ th-best sequence overlaps with an entity  $E_j$  extracted by the  $j$ th-best sequence, where  $i \leq j$ , then we discard  $E_j$ . If  $E_i$  and  $E_j$  do not overlap, then we consider both entities.

by adding the relation extraction step, the pipeline baselines are able to improve precision over the CRF but fail at recall. CRF+Syn and CRF+Adj provide the same performance as CRF, since the relation extraction step only affects the results of opinion arguments. By incorporating syntactic information, CRF+Syn provides better precision than CRF+Adj on extracting arguments at the expense of recall. This indicates that using simple syntactic rules would mistakenly filter many correct relations. By using binary classifiers to predict relations, CRF+RE produces high precision on opinion and target extraction but also results in very low recall. Using the *exact* metric, we observe the same general trend in the results as the *overlap* metric. The scores are lower since the metric is much stricter.

Table 3 shows the results of opinion relation extraction using the *overlap* metric. We compare our approach with pipelined baselines in two settings: one employs relation extraction on 1-best output of CRF (top half of table) and the other employs the merged 10-best output of CRF (bottom half of table). We can see that in general, using merged 10-best CRF outputs boosts the recall while sacrificing precision. This is expected since merging the 10-best CRF outputs favors candidates that are

Method	IS-ABOUT Relation Extraction			IS-FROM Relation Extraction		
	P	R	F1	P	R	F1
ILP-W/O-ENTITY	49.10	40.48	44.38	44.77	58.24	50.63
ILP-W-SINGLE-RE	63.88	49.35	55.68	53.64	65.02	58.78
ILP-W/O-IMPLICIT-RE	62.00	44.73	51.97	73.23	51.28	60.32
Joint-Model	64.38	51.20	<b>57.04**</b>	64.97	58.61	<b>61.63*</b>

Table 4: Comparison between our approach and ILP baselines that omit some potentials in our approach.

believed to be more accurate by the CRF predictor. If CRF makes mistakes, the mistakes will propagate to the relation extraction step. The poor performance on precision further confirms the error propagation problem in the pipeline approaches. In contrast, our joint-inference method successfully boosts the recall while maintaining reasonable precision. This demonstrates that joint inference can effectively leverage the advantage of individual predictors and limit error propagation.

To demonstrate the effectiveness of different potentials in our joint inference model, we consider three variants of our ILP formulation that omit some potentials in the joint inference: one is ILP-W/O-ENTITY, which extracts opinion relations without integrating information from opinion entity identification; one is ILP-W-SINGLE-RE, which focuses on extracting a single opinion relation and ignores the information from the other relation; the third one is ILP-W/O-IMPLICIT-RE, which omits the potential for *opinion-implicit-arg* relation and assumes every opinion expression is linked to an explicit argument. The objective function of ILP-W/O-ENTITY can be represented as

$$\arg \max_u \sum_k \sum_{i \in \mathcal{O}} \sum_{j \in \mathcal{A}_k} r_{ij} u_{ij} \quad (2)$$

which is subject to constraints on  $u_{ij}$  to enforce relations to not overlap and limit the maximum number of relations that can be extracted for each opinion expression and each argument. For ILP-W-SINGLE-RE, we simply remove the variables associated with one opinion relation in the objective function (1) and constraints. The formulation of ILP-W/O-IMPLICIT-RE removes the variables associated with potential  $r_i$  in the objective function and corresponding constraints. It can be viewed as an extension to the ILP approach in Choi et al. (2006) that includes opinion targets and uses simpler ILP formulation with only one parameter and fewer binary variables and constraints to represent entity label assignments<sup>11</sup>.

<sup>11</sup>We compared the proposed ILP formulation with the ILP

Table 4 shows the results of these methods on opinion relation extraction. We can see that without the knowledge of the entity extractor, ILP-W/O-ENTITY provides poor performance on both relation extraction tasks. This confirms the effectiveness of leveraging knowledge from entity extractor and relation extractor. The improvement yielded by our approach over ILP-W-SINGLE-RE demonstrates the benefit of jointly optimizing different types of opinion relations. Our approach also outperforms ILP-W/O-IMPLICIT-RE, which does not take into account implicit relations. The results demonstrate that incorporating knowledge of implicit opinion relations is important.

## 6 Discussion

We note that the joint inference model yields a clear improvement on recall but not on precision compared to the CRF-based baselines. Analyzing the errors, we found that the joint model extracts comparable number of opinion entities compared to the gold standard, while the CRF-based baselines extract significantly fewer opinion entities (around 60% of the number of entities in the gold standard). With more extracted opinion entities, the precision is sacrificed but recall is boosted substantially, and overall we see an increase in F-measure. We also found that a good portion of errors were made because the generated candidates failed to cover the correct solutions. Recall that the joint model finds the global optimal solution over a set of opinion entity and relation candidates, which are obtained from the n-best CRF predictions and constituents in the parse tree that satisfy certain syntactic patterns. It is possible that the generated candidates do not contain the gold standard answers. For example, our model failed to identify the IS-ABOUT relation (*offers, general aid*) from the following sentence *Powell had contacted ... and received offers<sub>O1</sub> of [gen-*

formulation in Choi et al. (2006) on extracting opinion holders, opinion expressions and IS-FROM relations, and showed that the proposed ILP formulation performs better on all three extraction tasks.

eral aid] $T_1$ ... because both the CRF predictor and syntactic heuristics fail to capture (*offers, general aid*) as a potential relation candidate. By applying simple heuristics such as treating all verbs or verb phrases as opinion candidates would not help because it would introduce a large number of negative candidates and lower the accuracy of relation extraction (only 52% of the opinion expressions are verbs or verb phrases and 64% of the opinion targets are noun or noun phrases in the corpus we used). Therefore a more effective candidate generation method is needed to allow more candidates while limiting the number of negative candidates. We also observed incorrect parsing to be a cause of error. We hope to study ways to account for such errors in our approach as future work.

For computational time, our ILP formulation can be solved very efficiently using advanced ILP solvers. In our experiment, using GLPK's branch-and-cut solver took 0.2 seconds to produce optimal ILP solutions for 1000 sentences on a machine with Intel Core 2 Duo CPU and 4GB RAM.

## 7 Conclusion

In this paper we propose a joint inference approach for extracting opinion-related entities and opinion relations. We decompose the task into different subproblems, and jointly optimize them using constraints that aim to encourage their consistency and reduce prediction uncertainty. We show that our approach can effectively integrate knowledge from different predictors and achieve significant improvements in overall performance for opinion extraction. For future work, we plan to extend our model to handle more complex opinion relations, e.g. nesting or cross-sentential relations. This can be potentially addressed by incorporating more powerful predictors and more complex linguistic constraints.

## Acknowledgments

This work was supported in part by DARPA-BAA-12-47 DEFT grant 12475008 and NSF grant BCS-0904822. We thank Igor Labutov for helpful discussion and suggestions, Ainur Yessenalina for early discussion of the work, as well as the reviews for helpful comments.

## References

- E. Breck, Y. Choi, and C. Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of the 20th international joint conference on Artificial intelligence*, pages 2683–2688. Morgan Kaufmann Publishers Inc.
- Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 355–362. Association for Computational Linguistics.
- Y. Choi, E. Breck, and C. Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 431–439. Association for Computational Linguistics.
- D. Das, A.F.T. Martins, and N.A. Smith. 2012. An exact dual decomposition algorithm for shallow semantic parsing with constraints. *Proceedings of\* SEM.[ii, 10, 50]*.
- Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.
- M. Hu and B. Liu. 2004. Mining opinion features in customer reviews. In *Proceedings of the National Conference on Artificial Intelligence*, pages 755–760. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.
- Richard Johansson and Alessandro Moschitti. 2010a. Reranking models in fine-grained opinion analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 519–527. Association for Computational Linguistics.
- Richard Johansson and Alessandro Moschitti. 2010b. Syntactic and semantic structure for opinion expression detection. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 67–76. Association for Computational Linguistics.
- Richard Johansson and Alessandro Moschitti. 2012. Relational features in fine-grained opinion analysis.
- Soo-Min Kim and Eduard Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pages 1–8. Association for Computational Linguistics.
- N. Kobayashi, K. Inui, and Y. Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural*

- Language Learning (EMNLP-CoNLL)*, pages 1065–1074.
- John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
- K. Liu, L. Xu, and J. Zhao. 2012. Opinion target extraction using word-based translation model. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.
- Bo Pang and Lillian Lee. 2008. *Opinion mining and sentiment analysis*. Now Pub.
- V. Punyakanok, D. Roth, and W. Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international joint conference on Artificial intelligence*, pages 1199–1204. Morgan Kaufmann Publishers Inc.
- G. Qiu, B. Liu, J. Bu, and C. Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1):9–27.
- R. Quirk, S. Greenbaum, G. Leech, J. Svartvik, and D. Crystal. 1985. *A comprehensive grammar of the English language*, volume 397. Cambridge Univ Press.
- D. Roth and W. Yih. 2004. *A linear programming formulation for global inference in natural language tasks*. Defense Technical Information Center.
- J. Ruppenhofer, S. Somasundaran, and J. Wiebe. 2008. Finding the sources and targets of subjective expressions. In *Proceedings of LREC*.
- Vivek Srikumar and Dan Roth. 2011. A joint model for extended semantic role labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 129–139. Association for Computational Linguistics.
- V. Stoyanov and C. Cardie. 2008. Topic identification for fine-grained opinion analysis. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 817–824. Association for Computational Linguistics.
- J. Wiebe, T. Wilson, and C. Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2009. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational linguistics*, 35(3):399–433.
- Theresa Wilson. 2008. *Fine-Grained Subjectivity Analysis*. Ph.D. thesis, Ph. D. thesis, University of Pittsburgh. Intelligent Systems Program.
- Y. Wu, Q. Zhang, X. Huang, and L. Wu. 2009. Phrase dependency parsing for opinion mining. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1533–1541. Association for Computational Linguistics.
- B. Yang and C. Cardie. 2012. Extracting opinion expressions with semi-markov conditional random fields. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O’Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1462–1470. Association for Computational Linguistics.

# Linguistic Models for Analyzing and Detecting Biased Language

**Marta Recasens**  
Stanford University  
recasens@google.com

**Cristian Danescu-Niculescu-Mizil**  
Stanford University  
Max Planck Institute SWS  
cristiand@cs.stanford.edu

**Dan Jurafsky**  
Stanford University  
jurafsky@stanford.edu

## Abstract

Unbiased language is a requirement for reference sources like encyclopedias and scientific texts. Bias is, nonetheless, ubiquitous, making it crucial to understand its nature and linguistic realization and hence detect bias automatically. To this end we analyze real instances of human edits designed to remove bias from Wikipedia articles. The analysis uncovers two classes of bias: *framing bias*, such as praising or perspective-specific words, which we link to the literature on subjectivity; and *epistemological bias*, related to whether propositions that are presupposed or entailed in the text are uncontroversially accepted as true. We identify common linguistic cues for these classes, including factive verbs, implicatives, hedges, and subjective intensifiers. These insights help us develop features for a model to solve a new prediction task of practical importance: given a biased sentence, identify the bias-inducing word. Our linguistically-informed model performs almost as well as humans tested on the same task.

## 1 Introduction

Writers and editors of reference works such as encyclopedias, textbooks, and scientific articles strive to keep their language **unbiased**. For example, Wikipedia advocates a policy called *neutral point of view (NPOV)*, according to which articles should represent “fairly, proportionately, and as far as possible without bias, all significant views that have been published by reliable sources” (Wikipedia, 2013b). Wikipedia’s style guide asks editors to use nonjudgmental language, to indicate the relative prominence of opposing points of view, to avoid presenting uncontroversial

facts as mere opinion, and, conversely, to avoid stating opinions or contested assertions as facts.

Understanding the linguistic realization of bias is important for linguistic theory; automatically detecting these biases is equally significant for computational linguistics. We propose to address both by using a powerful resource: edits in Wikipedia that are specifically designed to remove bias. Since Wikipedia maintains a complete revision history, the edits associated with NPOV tags allow us to compare the text in its biased (before) and unbiased (after) form, helping us better understand the linguistic realization of bias. Our work thus shares the intuition of prior NLP work applying Wikipedia’s revision history (Nelken and Yamangil, 2008; Yatskar et al., 2010; Max and Wisniewski, 2010; Zanzotto and Pennacchiotti, 2010).

The analysis of Wikipedia’s edits provides valuable linguistic insights into the nature of biased language. We find two major classes of bias-driven edits. The first, **framing bias**, is realized by subjective words or phrases linked with a particular point of view. In (1), the term *McMansion*, unlike *homes*, appeals to a negative attitude toward large and pretentious houses. The second class, **epistemological bias**, is related to linguistic features that subtly (often via presupposition) focus on the believability of a proposition. In (2), the assertive *stated* removes the bias introduced by *claimed*, which casts doubt on Kuypers’ statement.

- (1) a. Usually, smaller cottage-style houses have been demolished to make way for these **McMansions**.  
b. Usually, smaller cottage-style houses have been demolished to make way for these **homes**.
- (2) a. Kuypers **claimed** that the mainstream press in America tends to favor liberal viewpoints.  
b. Kuypers **stated** that the mainstream press in America tends to favor liberal viewpoints.

Bias is linked to the lexical and grammatical cues identified by the literature on subjectivity (Wiebe et al., 2004; Lin et al., 2011), sentiment (Liu et al., 2005; Turney, 2002), and especially stance

or “arguing subjectivity” (Lin et al., 2006; Somasundaran and Wiebe, 2010; Yano et al., 2010; Park et al., 2011; Conrad et al., 2012). For example, like stance, framing bias is realized when the writer of a text takes a particular position on a controversial topic and uses its metaphors and vocabulary. But unlike the product reviews or debate articles that overtly use subjective language, editors in Wikipedia are actively trying to avoid bias, and hence biases may appear more subtly, in the form of covert framing language, or presuppositions and entailments that may not play as important a role in other genres. Our linguistic analysis identifies common classes of these subtle bias cues, including factive verbs, implicatives and other entailments, hedges, and subjective intensifiers.

Using these cues could help automatically detect and correct instances of bias, by first finding biased phrases, then identifying the word that introduces the bias, and finally rewording to eliminate the bias. In this paper we propose a solution for the second of these tasks, identifying the bias-inducing word in a biased phrase. Since, as we show below, this task is quite challenging for humans, our system has the potential to be very useful in improving the neutrality of reference works like Wikipedia. Tested on a subset of non-neutral sentences from Wikipedia, our model achieves 34% accuracy—and up to 59% if the top three guesses are considered—on this difficult task, outperforming four baselines and nearing humans tested on the same data.

## 2 Analyzing a Dataset of Biased Language

We begin with an empirical analysis based on Wikipedia’s bias-driven edits. This section describes the data, and summarizes our linguistic analysis.<sup>1</sup>

### 2.1 The NPOV Corpus from Wikipedia

Given Wikipedia’s strict enforcement of an NPOV policy, we decided to build the **NPOV corpus**, containing Wikipedia edits that are specifically designed to remove bias. Editors are encouraged to identify and rewrite biased passages to achieve a more neutral tone, and they can use several NPOV

<sup>1</sup>The data and bias lexicon we developed are available at [http://www.mpi-sws.org/~cristian/Biased\\_language.html](http://www.mpi-sws.org/~cristian/Biased_language.html)

Data	Articles	Revisions	Words	Edits	Sents
Train	5997	2238K	11G	13807	1843
Dev	653	210K	0.9G	1261	163
Test	814	260K	1G	1751	230
Total	7464	2708K	13G	16819	2235

Table 1: Statistics of the NPOV corpus, extracted from Wikipedia. (*Edits* refers to bias-driven edits, i.e., with an NPOV comment. *Sents* refers to sentences with a one-word bias-driven edit.)

tags to mark biased content.<sup>2</sup> Articles tagged this way fall into Wikipedia’s category of *NPOV disputes*.

We constructed the NPOV corpus by retrieving all articles that were or had been in the NPOV-dispute category<sup>3</sup> together with their full revision history. We used Stanford’s CoreNLP tools<sup>4</sup> to tokenize and split the text into sentences. Table 1 shows the statistics of this corpus, which we split into training (train), development (dev), and test. Following Wikipedia’s terminology, we call each version of a Wikipedia article a *revision*, and so an article can be viewed as a set of (chronologically ordered) revisions.

### 2.2 Extracting Edits Meant to Remove Bias

Given all the revisions of a page, we extracted the changes between pairs of revisions with the word-mode *diff* function from the Diff Match and Patch library.<sup>5</sup> We refer to these changes between revisions as *edits*, e.g., *McMansion > large home*. An edit consists of two strings: the old string that is being replaced (i.e., the before form), and the new modified string (i.e., the after form).

Our assumption was that among the edits happening in NPOV disputes, we would have a high density of edits intended to remove bias, which we call *bias-driven edits*, like (1) and (2) from Section 1. But many other edits occur even in NPOV disputes, including edits to fix spelling or grammatical errors, simplify the language, make the meaning more precise, or even vandalism (Max

<sup>2</sup>`{{POV}}`, `{{POV-check}}`, `{{POV-section}}`, etc. Adding these tags displays a template such as “The neutrality of this article is disputed. Relevant discussion may be found on the talk page. Please do not remove this message until the dispute is resolved.”

<sup>3</sup>[http://en.wikipedia.org/wiki/Category:All\\_NPOV\\_disputes](http://en.wikipedia.org/wiki/Category:All_NPOV_disputes)

<sup>4</sup><http://nlp.stanford.edu/software/corenlp.shtml>

<sup>5</sup><http://code.google.com/p/google-diff-match-patch>

and Wisniewski, 2010). Therefore, in order to extract a high-precision set of bias-driven edits, we took advantage of the comments that editors can associate with a revision—typically short and brief sentences describing the reason behind the revision. We considered as bias-driven edits those that appeared in a revision whose comment mentioned (*N*)*POV*, e.g., *Attempts at presenting some claims in more NPOV way*; or *merging in a passage from the researchers article after basic NPOV-ing*. We only kept edits whose before and after forms contained five or fewer words, and discarded those that only added a hyperlink or that involved a minimal change (character-based Levenshtein distance < 4). The final number of bias-driven edits for each of the data sets is shown in the “Edits” column of Table 1.

### 2.3 Linguistic Analysis

Style guides talk about biased language in a prescriptive manner, listing a few words that should be avoided because they are flattering, vague, or endorse a particular point of view (Wikipedia, 2013a). Our focus is on analyzing actual biased text and bias-driven edits extracted from Wikipedia.

As we suggested above, this analysis uncovered two major classes of bias: epistemological bias and framing bias. Table 2 shows the distribution (from a sample of 100 edits) of the different types and subtypes of bias presented in this section.

(A) **Epistemological bias** involves propositions that are either commonly agreed to be true or commonly agreed to be false and that are subtly presupposed, entailed, asserted or hedged in the text.

1. **Factive verbs** (Kiparsky and Kiparsky, 1970) presuppose the truth of their complement clause. In (3-a) and (4-a), *realize* and *reveal* presuppose the truth of “the oppression of black people...” and “the Meditation technique produces...”, whereas (3-b) and (4-b) present the two propositions as somebody’s stand or an experimental result.
  - (3) a. **He realized** that the oppression of black people was more of a result of economic exploitation than anything innately racist.
  - b. **His stand was** that the oppression of black people was more of a result of economic exploitation than anything innately racist.
  - (4) a. The first research **revealed** that the Meditation technique produces a unique state fact.
  - b. The first research **indicated** that the Meditation technique produces a unique state fact.

Bias	Subtype	%
A. Epistemological bias		43
	- Factive verbs	3
	- Entailments	25
	- Assertives	11
	- Hedges	4
B. Framing bias		57
	- Intensifiers	19
	- One-sided terms	38

Table 2: Proportion of the different bias types.

2. **Entailments** are directional relations that hold whenever the truth of one word or phrase follows from another, e.g., *murder* entails *kill* because there cannot be murdering without killing (5). However, *murder* entails killing in an unlawful, premeditated way. This class includes implicative verbs (Karttunen, 1971), which imply the truth or untruth of their complement, depending on the polarity of the main predicate. In (6-a), *coerced into accepting* entails accepting in an unwilling way.

- (5) a. After he **murdered** three policemen, the colony proclaimed Kelly a wanted outlaw.
- b. After he **killed** three policemen, the colony proclaimed Kelly a wanted outlaw.

- (6) a. A computer engineer who **was coerced into accepting** a plea bargain.
- b. A computer engineer who **accepted** a plea bargain.

3. **Assertive verbs** (Hooper, 1975) are those whose complement clauses assert a proposition. The truth of the proposition is not presupposed, but its level of certainty depends on the asserting verb. Whereas verbs of saying like *say* and *state* are usually neutral, *point out* and *claim* cast doubt on the certainty of the proposition.

- (7) a. The “no Boeing” theory is a controversial issue, even among conspiracists, many of whom have **pointed out** that it is disproved by ...
- b. The “no Boeing” theory is a controversial issue, even among conspiracists, many of whom have **said** that it is disproved by...

- (8) a. Cooper says that slavery was worse in South America and the US than Canada, but **clearly states** that it was a horrible and cruel practice.
- b. Cooper says that slavery was worse in South America and the US than Canada, but **points out** that it was a horrible and cruel practice.

4. **Hedges** are used to reduce one’s commitment to the truth of a proposition, thus avoiding any bold predictions (9-b) or statements (10-a).<sup>6</sup>

- (9) a. Eliminating the profit motive **will decrease** the rate of medical innovation.
- b. Eliminating the profit motive **may have a lower** rate of medical innovation.
- (10) a. The lower cost of living in more rural areas means a **possibly** higher standard of living.
- b. The lower cost of living in more rural areas means a higher standard of living.

Epistemological bias is bidirectional, that is, bias can occur because doubt is cast on a proposition commonly assumed to be true, or because a presupposition or implication is made about a proposition commonly assumed to be false. For example, in (7) and (8) above, *point out* is replaced in the former case, but inserted in the second case. If the truth of the proposition is uncontroversially accepted by the community (i.e., reliable sources, etc.), then the use of a factive is unbiased. In contrast, if only a specific viewpoint agrees with its truth, then using a factive is biased.

**(B) Framing bias** is usually more explicit than epistemological bias because it occurs when subjective or one-sided words are used, revealing the author’s stance in a particular debate (Entman, 2007).

1. **Subjective intensifiers** are adjectives or adverbs that add (subjective) force to the meaning of a phrase or proposition.

- (11) a. Schnabel himself did the **fantastic** reproductions of Basquiat’s work.
- b. Schnabel himself did the **accurate** reproductions of Basquiat’s work.
- (12) a. Shwekey’s albums are arranged by many **talented** arrangers.
- b. Shwekey’s albums are arranged by many **different** arrangers.

2. **One-sided terms** reflect only one of the sides of a contentious issue. They often belong to controversial subjects (e.g., religion, terrorism, etc.) where the same event can be seen from two or more opposing perspectives, like the Israeli-Palestinian conflict (Lin et al., 2006).

- (13) a. Israeli forces **liberated** the eastern half of Jerusalem.
- b. Israeli forces **captured** the eastern half of Jerusalem.
- (14) a. Concerned Women for America’s major areas of political activity have consisted of opposition to gay causes, **pro-life** law...
- b. Concerned Women for America’s major areas of political activity have consisted of opposition to gay causes, **anti-abortion** law...
- (15) a. Colombian **terrorist** groups.
- b. Colombian **paramilitary** groups.

Framing bias has been studied within the literature on stance recognition and arguing subjectivity. Because this literature has focused on identifying which side an article takes on a two-sided debate such as the Israeli-Palestinian conflict (Lin et al., 2006), most studies cast the problem as a two-way classification of documents or sentences into *for*/positive vs. *against*/negative (Anand et al., 2011; Conrad et al., 2012; Somasundaran and Wiebe, 2010), or into one of two opposing views (Yano et al., 2010; Park et al., 2011). The features used by these models include subjectivity and sentiment lexicons, counts of unigrams and bigrams, distributional similarity, discourse relationships, and so on.

The datasets used by these studies come from genres that overtly take a specific stance (e.g., debates, editorials, blog posts). In contrast, Wikipedia editors are asked not to advocate a particular point of view, but to provide a balanced account of the different available perspectives. For this reason, overtly biased opinion statements such as “I believe that...” are not common in Wikipedia. The features used by the subjectivity literature help us detect framing bias, but we also need features that capture epistemological bias expressed through presuppositions and entailments.

### 3 Automatically Identifying Biased Language

We now show how the bias cues identified in Section 2.3 can help solve a new task. Given a biased sentence (e.g., a sentence that a Wikipedia editor has tagged as violating the NPOV policy), our goal in this new task is to identify the word that introduces bias. This is part of a potential three-step process for detecting and correcting biased language: (1) finding biased phrases, (2) identifying the word that introduces the bias, (3) rewording to eliminate the bias. As we will see below, it can be

<sup>6</sup>See Choi et al. (2012) for an exploration of the interface between hedging and framing.

hard even for humans to track down the sources of bias, because biases in reference works are often subtle and implicit. An automatic bias detector that can highlight the bias-inducing word(s) and draw the editors’ attention to words that need to be modified could thus be important for improving reference works like Wikipedia or even in news reporting.

We selected the subset of sentences that had a single NPOV edit involving one (original) word. (Although the before form consists of only one word, the after form can be either one or more words or the null string (i.e., deletion edits); we do not use the after string in this identification task). The number of sentences in the train, dev and test sets is shown in the last column of Table 1.

We trained a logistic regression model on a feature vector for every word that appears in the NPOV sentences from the training set, with the bias-inducing words as the positive class, and all the other words as the negative class. The features are described in the next section.

At test time, the model is given a set of sentences and, for each of them, it ranks the words according to their probability to be biased, and outputs the highest ranked word (TOP1 model), the two highest ranked words (TOP2 model), or the three highest ranked words (TOP3 model).

### 3.1 Features

The types of features used in the logistic regression model are listed in Table 3, together with their value space. The total number of features is 36,787. The ones targeting framing bias draw on previous work on sentiment and subjectivity detection (Wiebe et al., 2004; Liu et al., 2005). Features to capture epistemological bias are based on the bias cues identified in Section 2.3.

A major split separates the features that describe the word under analysis (e.g., lemma, POS, whether it is a hedge, etc.) from those that describe its surrounding context (e.g., the POS of the word to the left, whether there is a hedge in the context, etc.). We define *context* as a 5-gram window, i.e., two words to the left of the word under analysis, and two to the right. Taking context into account is important given that biases can be context-dependent, especially epistemological bias since it depends on the truth of a proposition. To define some of the features like POS and grammatical relation, we used the Stanford’s CoreNLP

tagger and dependency parser (de Marneffe et al., 2006).

Features 9–10 use the list of hedges from Hyland (2005), features 11–14 use the factives and assertives from Hooper (1975), features 15–16 use the implicatives from Karttunen (1971), features 19–20 use the entailments from Berant et al. (2012), features 21–25 employ the subjectivity lexicon from Riloff and Wiebe (2003), and features 26–29 use the sentiment lexicon—positive and negative words—from Liu et al. (2005). If the word (or a word in the context) is in the lexicon, then the feature is true, otherwise it is false.

We also included a “bias lexicon” (feature 31) that we built based on our NPOV corpus from Wikipedia. We used the training set to extract the lemmas of words that were the before form of at least two NPOV edits, and that occurred in at least two different articles. Of the 654 words included in this lexicon, 433 were unique to this lexicon (i.e., recorded in neither Riloff and Wiebe’s (2003) subjectivity lexicon nor Liu et al.’s (2005) sentiment lexicon) and represented many one-sided or controversial terms, e.g., *abortion*, *same-sex*, *execute*.

Finally, we also included a “collaborative feature” that, based on the previous revisions of the edit’s article, computes the ratio between the number of times that the word was NPOV-edited and its frequency of occurrence. This feature was designed to capture framing bias specific to an article or topic.

### 3.2 Baselines

Previous work on subjectivity and stance recognition has been evaluated on the task of classifying documents as opinionated vs. factual, *for* vs. *against*, positive vs. negative. Given that the task of identifying the bias-inducing word of a sentence is novel, there were no previous results to compare directly against. We ran the following five baselines.

1. **Random guessing.** Naively returns a random word from every sentence.
2. **Role baseline.** Selects the word with the syntactic role that has the highest probability to be biased, as computed on the training set. This is the parse tree root (probability  $p = .126$  to be biased), followed by verbal arguments ( $p = .085$ ), and the subject ( $p = .084$ ).

ID	Feature	Value	Description
1*	Word	<string>	Word $w$ under analysis.
2	Lemma	<string>	Lemma of $w$ .
3*	POS	{NNP, JJ, ...}	POS of $w$ .
4*	POS - 1	{NNP, JJ, ...}	POS of one word before $w$ .
5	POS - 2	{NNP, JJ, ...}	POS of two words before $w$ .
6*	POS + 1	{NNP, JJ, ...}	POS of one word after $w$ .
7	POS + 2	{NNP, JJ, ...}	POS of two words after $w$ .
8	Position in sentence	{start, mid, end}	Position of $w$ in the sentence (split into three parts).
9	Hedge	{true, false}	$w$ is in Hyland’s (2005) list of hedges (e.g., <i>apparently</i> ).
10*	Hedge in context	{true, false}	One/two words around $w$ is a hedge (Hyland, 2005).
11*	Factive verb	{true, false}	$w$ is in Hooper’s (1975) list of factives (e.g., <i>realize</i> ).
12*	Factive verb in context	{true, false}	One/two word(s) around $w$ is a factive (Hooper, 1975).
13*	Assertive verb	{true, false}	$w$ is in Hooper’s (1975) list of assertives (e.g., <i>claim</i> ).
14*	Assertive verb in context	{true, false}	One/two word(s) around $w$ is an assertive (Hooper, 1975).
15	Implicative verb	{true, false}	$w$ is in Karttunen’s (1971) list of implicatives (e.g., <i>manage</i> ).
16*	Implicative verb in context	{true, false}	One/two word(s) around $w$ is an implicative (Karttunen, 1971).
17*	Report verb	{true, false}	$w$ is a report verb (e.g., <i>add</i> ).
18	Report verb in context	{true, false}	One/two word(s) around $w$ is a report verb.
19*	Entailment	{true, false}	$w$ is in Berant et al.’s (2012) list of entailments (e.g., <i>kill</i> ).
20*	Entailment in context	{true, false}	One/two word(s) around $w$ is an entailment (Berant et al., 2012).
21*	Strong subjective	{true, false}	$w$ is in Riloff and Wiebe’s (2003) list of strong subjectives (e.g., <i>absolute</i> ).
22	Strong subjective in context	{true, false}	One/two word(s) around $w$ is a strong subjective (Riloff and Wiebe, 2003).
23*	Weak subjective	{true, false}	$w$ is in Riloff and Wiebe’s (2003) list of weak subjectives (e.g., <i>noisy</i> ).
24*	Weak subjective in context	{true, false}	One/two word(s) around $w$ is a weak subjective (Riloff and Wiebe, 2003).
25	Polarity	{+, -, both, ...}	The polarity of $w$ according to Riloff and Wiebe (2003), e.g., <i>praising</i> is positive.
26*	Positive word	{true, false}	$w$ is in Liu et al.’s (2005) list of positive words (e.g., <i>excel</i> ).
27*	Positive word in context	{true, false}	One/two word(s) around $w$ is positive (Liu et al., 2005).
28*	Negative word	{true, false}	$w$ is in Liu et al.’s (2005) list of negative words (e.g., <i>terrible</i> ).
29*	Negative word in context	{true, false}	One/two word(s) around $w$ is negative (Liu et al., 2005).
30*	Grammatical relation	{root, subj, ...}	Whether $w$ is the subject, object, root, etc. of its sentence.
31	Bias lexicon	{true, false}	$w$ has been observed in NPOV edits (e.g., <i>nationalist</i> ).
32*	Collaborative feature	<numeric>	Number of times that $w$ was NPOV-edited in the article’s prior history / frequency of $w$ .

Table 3: Features used by the bias detector. The star (\*) shows the most contributing features.

3. **Sentiment baseline.** Logistic regression model that only uses the features based on Liu et al.’s (2005) lexicons of positive and negative words (i.e., features 26–29).

4. **Subjectivity baseline.** Logistic regression model that only uses the features based on Riloff and Wiebe’s (2003) lexicon of subjective words (i.e., features 21–25).

5. **Wikipedia baseline.** Selects as biased the words that appear in Wikipedia’s list of words to avoid (Wikipedia, 2013a).

These baselines assessed the difficulty of the task, as well as the extent to which traditional sentiment-analysis and subjectivity features would suffice to detect biased language.

### 3.3 Results and Discussion

To measure performance, we used accuracy defined as:

$$\frac{\text{\#sentences with the correctly predicted biased word}}{\text{\#sentences}}$$

The results are shown in Table 4. As explained earlier, we evaluated all the models by outputting as biased either the highest ranked word or the two or three highest ranked words. These correspond to the TOP1, TOP2 and TOP3 columns, respectively. The TOP3 score increases to 59%. A tool that highlights up to three words to be revised would simplify the editors’ job and decrease significantly the time required to revise.

Our model outperforms all five baselines by a large margin, showing the importance of considering a wide range of features. Wikipedia’s list of words to avoid falls very short on recall. Fea-

System	TOP1	TOP2	TOP3
Baseline 1: Random	2.18	7.83	9.13
Baseline 2: Role	15.65	20.43	25.65
Baseline 3: Sentiment	14.78	22.61	27.83
Baseline 4: Subjectivity	16.52	25.22	33.91
Baseline 5: Wikipedia	10.00	10.00	10.00
Our system	<b>34.35</b>	<b>46.52</b>	<b>58.70</b>
Humans (AMT)	37.39	50.00	59.13

Table 4: Accuracy (%) of the bias detector on the test set.

tures that contribute the most to the model’s performance (in a feature ablation study on the dev set) are highlighted with a star (\*) in Table 3. In addition to showing the importance of linguistic cues for different classes of bias, the ablation study highlights the role of contextual features. The bias lexicon does not seem to help much, suggesting that it is overfit to the training data.

An error analysis shows that our system makes acceptable errors in that words wrongly predicted as bias-inducing may well introduce bias in a different context. In (16), the system picked *eschew*, whereas *orthodox* would have been the correct choice according to the gold edit. Note that both the sentiment and the subjectivity lexicons list *eschew* as a negative word. The bias type that poses the greatest challenge to the system are terms that are one-sided or loaded in a particular topic, such as *orthodox* in this example.

- (16) a. Some Christians *eschew* **orthodox** theology; such as the Unitarians, Socinian, [...]  
 b. Some Christians *eschew* **mainstream trinitarian** theology; such as the Unitarians, Socinian, [...]

The last row in Table 4 lists the performance of humans on the same task, presented in the next section.

## 4 Human Perception of Biased Language

Is it difficult for humans to find the word in a sentence that induces bias, given the subtle, often implicit biases in Wikipedia. We used Amazon Mechanical Turk<sup>7</sup> (AMT) to elicit annotations from humans for the same 230 sentences from the test set that we used to evaluate the bias detector in Section 3.3. The goal of this annotation was twofold: to compare the performance of our bias detector against a human baseline, and to assess the difficulty of this task for humans. While AMT labelers are not trained Wikipedia editors, under-

<sup>7</sup><http://www.mturk.com>

standing how difficult these cases are for untrained labelers is an important baseline.

### 4.1 Task

Our HIT (Human Intelligence Task) was called “Find the biased word!”. We kept the task description succinct. Turkers were shown Wikipedia’s definition of a “biased statement” and two example sentences that illustrated the two types of bias, framing and epistemological. In each HIT, annotators saw 10 sentences, one after another, and each one followed by a text box entitled “Word introducing bias.” For each sentence, they were asked to type in the text box the word that caused the statement to be biased. They were only allowed to enter a single word.

Before the 10 sentences, turkers were asked to list the languages they spoke as well as their primary language in primary school. This was English in all the cases. In addition, we included a probe question in the form of a paraphrasing task: annotators were given a sentence and two paraphrases (a correct and a bad one) to choose from. The goal of this probe question was to discard annotators who were not paying attention or did not have a sufficient command of English. This simple test was shown to be effective in verifying and eliciting linguistic attentiveness (Munro et al., 2010). This was especially important in our case as we were interested in using the human annotations as an oracle. At the end of the task, participants were given the option to provide additional feedback.

We split the 230 sentences into 23 sets of 10 sentences, and asked for 10 annotations of each set. Each approved HIT was rewarded with \$0.30.

### 4.2 Results and Discussion

On average, it took turkers about four minutes to complete each HIT. The feedback that we got from some of them confirmed our hypothesis that finding the bias source is difficult: “Some of the ‘biases’ seemed very slight if existent at all,” “This was a lot harder than I thought it would be... Interesting though!”.

We postprocessed the answers ignoring case, punctuation signs, and spelling errors. To ensure an answer quality as high as possible, we only kept those turkers who answered attentively by applying two filters: we only accepted answers that matched a valid word from the sentence, and we discarded answers from participants who did not

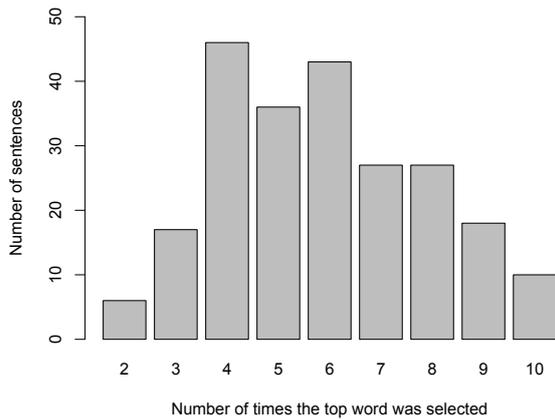


Figure 1: Distribution of the number of turkers who selected the top word (i.e., the word selected by the majority of turkers).

pass the paraphrasing task—there were six such cases. These filters provided us with confidence in the turkers’ answers as a fair standard of comparison.

Overall, humans correctly identified the biased word 30% of the time. For each sentence, we ranked the words according to the number of turkers (out of 10) who selected them and, like we did for the automated system, we assessed performance when considering only the top word (TOP1), the top 2 words (TOP2), and the top 3 words (TOP3). The last row of Table 4 reports the results. Only 37.39% of the majority answers coincided with the gold label, slightly higher than our system’s accuracy. The fact that the human answers are very close to the results of our system reflects the difficulty of the task. Biases in reference works can be very subtle and go unnoticed by humans; automated systems could thus be extremely helpful.

As a measure of inter-rater reliability, we computed pairwise agreement. The turkers agreed 40.73% of the time, compared to the 5.1% chance agreement that would be achieved if raters had randomly selected a word for each sentence. Figure 1 plots the number of times the top word of each sentence was selected. The bulk of the sentences only obtained between four and six answers for the same word.

There is a good amount of overlap (~34%) between the correct answers predicted by our system and those from humans. Much like the automated system, humans also have the hardest time identifying words that are one-sided or controversial to

a specific topic. They also picked *eschew* for (16) instead of *orthodox*. Compared to the system, they do better in detecting bias-inducing intensifiers, and about the same with epistemological bias.

## 5 Related Work

The work in this paper builds upon prior work on subjectivity detection (Wiebe et al., 2004; Lin et al., 2011; Conrad et al., 2012) and stance recognition (Yano et al., 2010; Somasundaran and Wiebe, 2010; Park et al., 2011), but applied to the genre of reference works such as Wikipedia. Unlike the blogs, online debates and opinion pieces which have been the major focus of previous work, bias in reference works is undesirable. As a result, the expression of bias is more implicit, making it harder to detect by both computers and humans. Of the two classes of bias that we uncover, *framing bias* is indeed strongly linked to subjectivity, but *epistemological bias* is not. In this respect, our research is comparable to Greene and Resnik’s (2009) work on identifying *implicit* sentiment or perspective in journalistic texts, based on semantico-syntactic choices.

Given that the data that we use is not supposed to be opinionated, our task consists in detecting (implicit) bias instead of classifying into side A or B documents about a controversial topic like ObamaCare (Conrad et al., 2012) or the Israeli-Palestinian conflict (Lin et al., 2006; Greene and Resnik, 2009). Our model detects whether all the relevant perspectives are fairly represented by identifying statements that are one-sided. To this end, the features based on subjectivity and sentiment lexicons turn out to be helpful, and incorporating more features for stance detection is an important direction for future work.

Other aspects of Wikipedia structure have been used for other NLP applications. The Wikipedia revision history has been used for spelling correction, text summarization (Nelken and Yamangil, 2008), lexical simplification (Yatskar et al., 2010), paraphrasing (Max and Wisniewski, 2010), and textual entailment (Zanzotto and Pennacchiotti, 2010). Ganter and Strube (2009) have used Wikipedia’s weasel-word tags to train a hedge detector. Callahan and Herring (2011) have examined cultural bias based on Wikipedia’s NPOV policy.

## 6 Conclusions

Our study of bias in Wikipedia has implications for linguistic theory and computational linguistics. We show that bias in reference works falls broadly into two classes, framing and epistemological. The cues to framing bias are more explicit and are linked to the literature on subjectivity; cues to epistemological bias are subtle and implicit, linked to presuppositions and entailments in the text. Epistemological bias has not received much attention since it does not play a major role in overtly opinionated texts, the focus of much research on stance recognition. However, our logistic regression model reveals that epistemological and other features can usefully augment the traditional sentiment and subjectivity features for addressing the difficult task of identifying the bias-inducing word in a biased sentence.

Identifying the bias-inducing word is a challenging task even for humans. Our linguistically-informed model performs nearly as well as humans tested on the same task. Given the subtlety of some of these biases, an automated system that highlights one or more potentially biased words would provide a helpful tool for editors of reference works and news reports, not only making them aware of unnoticed biases but also saving them hours of time. Future work could investigate the incorporation of syntactic features or further features from the stance detection literature. Features from the literature on veridicality (de Marneffe et al., 2012) could be informative of the writer’s commitment to the truth of the events described, and document-level features could help assess the extent to which the article provides a balanced account of all the facts and points of view.

Finally, the NPOV data and the bias lexicon that we release as part of this research could prove useful in other bias related tasks.

## Acknowledgments

We greatly appreciate the support of Jean Wu and Christopher Potts in running our task on Amazon Mechanical Turk, and all the Amazon Turkers who participated. We benefited from comments by Valentin Spitkovsky on a previous draft and from the helpful suggestions of the anonymous reviewers. The first author was supported by a Beatriu de Pinós postdoctoral scholarship (2010 BP-A00149) from Generalitat de Catalunya. The sec-

ond author was supported by NSF IIS-1016909. The last author was supported by the Center for Advanced Study in the Behavioral Sciences at Stanford.

## References

- Pranav Anand, Marilyn Walker, Rob Abbott, Jean E. Fox Tree, Robeson Bowmani, and Michael Minor. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of ACL-HLT 2011 Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pages 1–9.
- Jonathan Berant, Ido Dagan, Meni Adler, and Jacob Goldberger. 2012. Efficient tree-based approximation for entailment graph learning. In *Proceedings of ACL 2012*, pages 117–125.
- Ewa Callahan and Susan C. Herring. 2011. Cultural bias in Wikipedia articles about famous persons. *Journal of the American Society for Information Science and Technology*, 62(10):1899–1915.
- Eunsol Choi, Chenhao Tan, Lillian Lee, Cristian Danescu-Niculescu-Mizil, and Jennifer Spindel. 2012. Hedge detection as a lens on framing in the GMO debates: a position paper. In *Proceedings of the ACL-2012 Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics*, pages 70–79.
- Alexander Conrad, Janyce Wiebe, and Rebecca Hwa. 2012. Recognizing arguing subjectivity and argument tags. In *Proceedings of ACL-2012 Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics*, pages 80–88.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC 2006*.
- Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2012. Did it happen? The pragmatic complexity of veridicality assessment. *Computational Linguistics*, 38(2):301–333.
- Robert M. Entman. 2007. Framing bias: Media in the distribution of power. *Journal of Communication*, 57(1):163–173.
- Viola Ganter and Michael Strube. 2009. Finding hedges by chasing weasels: Hedge detection using Wikipedia tags and shallow linguistic features. In *Proceedings of ACL-IJCNLP 2009*, pages 173–176.
- Stephan Greene and Philip Resnik. 2009. More than words: Syntactic packaging and implicit sentiment. In *Proceedings of NAACL-HLT 2009*, pages 503–511.

- Joan B. Hooper. 1975. On assertive predicates. In J. Kimball, editor, *Syntax and Semantics*, volume 4, pages 91–124. Academic Press, New York.
- Ken Hyland. 2005. *Metadiscourse: Exploring Interaction in Writing*. Continuum, London and New York.
- Lauri Karttunen. 1971. Implicative verbs. *Language*, 47(2):340–358.
- Paul Kiparsky and Carol Kiparsky. 1970. Fact. In M. Bierwisch and K. E. Heidolph, editors, *Progress in Linguistics*, pages 143–173. Mouton, The Hague.
- Wei-Hao Lin, Theresa Wilson, Janyce Wiebe, and Alexander Hauptmann. 2006. Which side are you on? Identifying perspectives at the document and sentence levels. In *Proceedings of CoNLL 2006*, pages 109–116.
- Chenghua Lin, Yulan He, and Richard Everson. 2011. Sentence subjectivity detection with weakly-supervised learning. In *Proceedings of AFNLP 2011*, pages 1153–1161.
- Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion Observer: analyzing and comparing opinions on the Web. In *Proceedings of WWW 2005*, pages 342–351.
- Aurélien Max and Guillaume Wisniewski. 2010. Mining naturally-occurring corrections and paraphrases from Wikipedia’s revision history. In *Proceedings of LREC 2010*, pages 3143–3148.
- Robert Munro, Steven Bethard, Victor Kuperman, Vicky Tzuyin Lai, Robin Melnick, Christopher Potts, Tyler Schnoebelen, and Harry Tily. 2010. Crowdsourcing and language studies: the new generation of linguistic data. In *Proceedings of the NAACL-HLT 2010 Workshop on Creating Speech and Language Data With Amazons Mechanical Turk*, pages 122–130.
- Rani Nelken and Elif Yamangil. 2008. Mining Wikipedias article revision history for training Computational Linguistics algorithms. In *Proceedings of the 1st AAAI Workshop on Wikipedia and Artificial Intelligence*.
- Souneil Park, KyungSoon Lee, and Junehwa Song. 2011. Contrasting opposing views of news articles on contentious issues. In *Proceedings of ACL 2011*, pages 340–349.
- Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of EMNLP 2003*, pages 105–112.
- Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL-HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 116–124.
- Peter D. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL 2002*, pages 417–424.
- Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Computational Linguistics*, 30(3):277–308.
- Wikipedia. 2013a. Wikipedia: Manual of style / Words to watch. [http://en.wikipedia.org/wiki/Wikipedia:Words\\_to\\_avoid](http://en.wikipedia.org/wiki/Wikipedia:Words_to_avoid). [Retrieved February 5, 2013].
- Wikipedia. 2013b. Wikipedia: Neutral point of view. [http://en.wikipedia.org/wiki/Wikipedia:Neutral\\_point\\_of\\_view](http://en.wikipedia.org/wiki/Wikipedia:Neutral_point_of_view). [Retrieved February 5, 2013].
- Tae Yano, Philip Resnik, and Noah A. Smith. 2010. Shedding (a thousand points of) light on biased language. In *Proceedings of the NAACL-HLT 2010 Workshop on Creating Speech and Language Data With Amazons Mechanical Turk*, pages 152–158.
- Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Proceedings of NAACL-HLT 2010*, pages 365–368.
- Fabio M. Zanzotto and Marco Pennacchiotti. 2010. Expanding textual entailment corpora from Wikipedia using co-training. In *Proceedings of the 2nd Coling Workshop on The People’s Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 28–36.

# Evaluating a City Exploration Dialogue System Combining Question-Answering and Pedestrian Navigation

Srinivasan Janarthanam<sup>1</sup>, Oliver Lemon<sup>1</sup>, Phil Bartie<sup>2</sup>, Tiphaine Dalmás<sup>2</sup>,  
Anna Dickinson<sup>2</sup>, Xingkun Liu<sup>1</sup>, William Mackaness<sup>2</sup>, and Bonnie Webber<sup>2</sup>

<sup>1</sup> The Interaction Lab, Heriot-Watt University

<sup>2</sup> Edinburgh University

sc445@hw.ac.uk

## Abstract

We present a city navigation and tourist information mobile dialogue app with integrated question-answering (QA) and geographic information system (GIS) modules that helps pedestrian users to navigate in and learn about urban environments. In contrast to existing mobile apps which treat these problems independently, our Android app addresses the problem of navigation and touristic question-answering in an integrated fashion using a shared dialogue context. We evaluated our system in comparison with Samsung S-Voice (which interfaces to Google navigation and Google search) with 17 users and found that users judged our system to be significantly more interesting to interact with and learn from. They also rated our system above Google search (with the Samsung S-Voice interface) for tourist information tasks.

## 1 Introduction

We present a mobile dialogue system (an Android app) called **Spacebook** that addresses the problem of pedestrian navigation and tourist information in urban environments. There has been little prior work that addresses these two problems - navigation and tourist information provision - in an integrated way. By *navigation*, we refer to the problem of finding appropriate destinations to go to and the task of wayfinding to reach them and by *tourist information provision* we refer to the problem of meeting the informational needs of a user about entities such as museums, statues and famous personalities. A dialogue system such as this could serve as a personal tour guide to pedestrian tourists as they walk around unknown cities. With the proliferation of smartphones, there has been a

number of mobile apps developed to address these problems. However these apps have the following problems: first, they demand the user's visual attention because they predominantly present information on a mobile screen. This can be dangerous in urban environments, as well as being distracting. Second, these apps address the problems of navigation and tourist information independently and therefore do not have a shared interaction context. This means that users cannot switch between information and navigation tasks in a natural and fluid manner.

User1: Take me to the National Museum.
System2: The National Museum is about 300m away..
System3: At the KFC, turn left on to South Bridge
<b>System4</b> : Near you is the statue of David Hume.
User2: Who is David Hume.
System5: David Hume was a Scottish philosopher....
User3: Tell me more about David Hume.
System6: He was one of the most important figures in..
System7: You should be able to see <b>the museum</b> ...
User4: Tell me more about <b>the museum</b> .
System8: <b>The National Museum of Scotland</b> is a....

Table 1: An example interaction with the evaluated system

In contrast to many existing mobile apps, Spacebook has a speech-only interface and addresses both problems in an integrated way. We conjecture that with a speech-only interface, users can immerse themselves in exploring the city, and that because of the shared context they can switch between navigation and tourist information tasks more easily. Using the navigational context, Spacebook pushes point-of-interest information which can then initiate tourist information tasks using the QA module. Table 1 presents an example interaction with our system showing the integrated use of navigation and question-answering capabil-

ities. Utterances *System4-8* show the system’s capability to push information about nearby points-of-interest (PoI) during a navigation task and answer followup questions using the QA system (in utterances *User2* and *User3*). The final 3 utterances show a natural switch between navigation to an entity and QA about that entity.

We investigate whether our system using a combination of geographical information system (GIS) and natural language processing (NLP) technologies would be a better companion to pedestrian city explorers than the current state-of-the-art mobile apps. We hypothesize that, (1) users will find our speech-only interface to navigation efficient as it allows them to navigate without having to repeatedly look at a map and (2), that users will find a dialogue interface which integrates touristic question-answering and navigation within a shared context to be useful for finding information about entities in the urban environment. We first present some related work in section 2. We describe the architecture of the system in section 3. We then present our experimental design, results and analysis in sections 5, 6 and 7.

## 2 Related work

Mobile apps such as Siri, Google Maps Navigation, Sygic, etc. address the problem of navigation while apps like Triposo, Guidepal, Wikihood, etc. address the problem of tourist information by presenting the user with descriptive information about various points of interest (PoI) in the city. While some exploratory apps present snippets of information about a precompiled list of PoIs, other apps dynamically generate a list of PoIs arranged based on their proximity to the users. Users can also obtain specific information about PoIs using Search apps. Also, since these navigation and exploratory/search apps do not address both problems in an integrated way, users need to switch between them and therefore lose interaction context.

While most apps address these two problems independently, some like Google Now, Google Field Trip, etc, mix navigation with exploration. But such apps present information primarily visually on the screen for the user to read. Some of these are available for download at the Google Play Android app store<sup>1</sup>. Several dialogue and natural language systems have addressed the issue

<sup>1</sup><https://play.google.com/store>

of pedestrian navigation (Malaka and Zipf, 2000; Raubal and Winter, 2002; Dale et al., 2003; Bartie and Mackaness, 2006; Shroder et al., 2011; Dethlefs and Cuayáhuitl, 2011). There has also been recent interest in shared tasks for generating navigation instructions in indoor and urban environments (Byron et al., 2007; Janarthanam and Lemon, 2011). Some dialogue systems deal with presenting information concerning points of interest (Ko et al., 2005; Kashioka et al., 2011) and interactive question answering (Webb and Webber, 2009).

In contrast, Spacebook has the objective of keeping the user’s cognitive load low and preventing users from being distracted (perhaps dangerously so) from walking in the city (Kray et al., 2003). Also, it allows users to interleave the two sub-tasks seamlessly and can keep entities discussed in both tasks in shared context (as shown in Table 1).

## 3 Architecture

The architecture of the Spacebook system is shown in figure 1. Our architecture brings together Spoken Dialogue Systems (SDS), Geographic Information Systems (GIS) and Question-Answering (QA) technologies (Janarthanam et al., 2012). Its essentially a spoken dialogue system (SDS) consisting of an automatic speech recogniser (ASR), a semantic parser, an Interaction Manager, an utterance generator and a text-to-speech synthesizer (TTS). The GIS modules in this architecture are the City Model, the Visibility Engine, and the Pedestrian tracker. Users communicate with the system using a smartphone-based client app (an Android app) that sends users’ position, pace rate, and spoken utterances to the system, and delivers synthesised system utterances to the user.

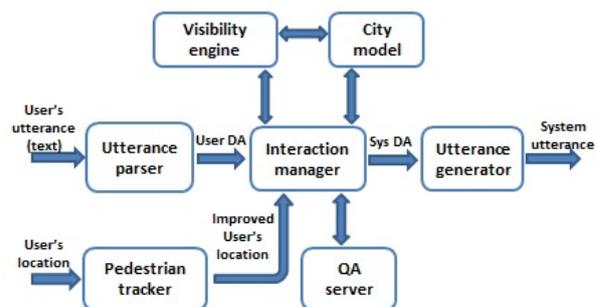


Figure 1: System Architecture

### 3.1 Dialogue interface

The dialogue interface consists of a speech recognition module, an utterance parser, an interaction manager, an utterance generator and a speech synthesizer. The Nuance 9 speech recogniser with a domain specific language model was used for speech recognition. The recognised speech is currently parsed using a rule-based parser into dialogue acts and semantic content.

The Interaction Manager (IM) is the central component of this architecture, which provides the user with navigational instructions, pushes PoI information and manages QA questions. It receives the user's input in the form of a dialogue act (DA), the user's location (latitude and longitude) and pace rate. Based on these inputs and the dialogue context, it responds with system output dialogue act, based on a dialogue policy. The IM initiates the conversation with a calibration phase where the user's initial location and orientation are obtained. The user can then initiate tasks that interest him/her. These tasks include searching for an entity (e.g. a museum or a restaurant), requesting navigation instructions to a destination, asking questions about the entities in the City Model, and so on. When the user is mobile, the IM identifies points of interest<sup>2</sup> on the route proximal to the user. We call this "PoI push". The user is encouraged to ask for more information if he/she is interested. The system also answers adhoc questions from the user (e.g. "Who is David Hume?", "What is the Old College?", etc) (see section 3.4).

Navigation instructions are given in-situ by observing user's position continuously, in relation to the next node (street junction) on the current planned route, and they are given priority if in conflict with a PoI push at the same time. Navigation instructions use landmarks near route nodes whenever possible (e.g. "When you reach Clydesdale Bank, keep walking forward"). The IM also informs when users pass by recognisable landmarks, just to reassure them that they are on track (e.g. "You will pass by Tesco on the right"). In addition to navigation instructions, the IM also answers users' questions concerning the route, his/her location, and location of and distance to the various entities. Finally, the IM uses the city model's Visibility Engine (VE) to determine whether the destination is visible to the user (see section 3.3).

<sup>2</sup>Using high scoring ones when there are many, based on tourist popularity ratings in the City Model.

The shared spatial and dialogue context employs a feature-based representation which is updated every 1 second (for location), and after every dialogue turn. Spatial context such as the user's coordinates, street names, PoIs and landmarks proximal to the user, etc are used by PoI pushing and navigation. The dialogue context maintains the history of landmarks and PoIs pushed, latest entities mentioned, etc to resolve anaphoric references in navigation and QA requests, and to deliver coherent dialogue responses. The IM resolves anaphoric references by keeping a record of entities mentioned in the dialogue context. It also engages in clarification sub-dialogues when the speech recognition confidence scores are low. The IM stores the name and type information for each entity (such as landmark, building, etc) mentioned in navigation instructions and PoI pushes. Subsequent references to these entities using expressions such as "the museum", "the cafe" etc are resolved by searching for the latest entity of the given type. Pronouns are resolved to the last mentioned entity.

The IM also switches between navigation, PoI push, and QA tasks in an intelligent manner by using the shared context to prioritise its utterances from these different tasks. The utterance generator is a Natural Language Generation module that translates the system DA into surface text which is converted into speech using the Cereproc Text-to-Speech Synthesizer using a Scottish female voice. The only changes made were minor adjustments to the pronunciation of certain place names.

### 3.2 Pedestrian tracker

Urban environments can be challenging with limited sky views, and hence limited line of sight to satellites, in deep urban corridors. There is therefore significant uncertainty about the user's true location reported by GNSS sensors on smartphones (Zandbergen and Barbeau, 2011). This module improves on the reported user position by combining smartphone sensor data (e.g. accelerometer) with map matching techniques, to determine the most likely location of the pedestrian (Bartie and Mackaness, 2012).

### 3.3 City Model

The City Model is a spatial database containing information about thousands of entities in the city of Edinburgh (Bartie and Mackaness, 2013). This data has been collected from a variety of exist-

ing resources such as Ordnance Survey, OpenStreetMap, Google Places, and the Gazetteer for Scotland. It includes the location, use class, name, street address, and where relevant other properties such as build date and tourist ratings. The model also includes a pedestrian network (streets, pavements, tracks, steps, open spaces) which is used by an embedded route planner to calculate minimal cost routes, such as the shortest path. The city model also consists of a Visibility Engine that identifies the entities that are in the user's *vista space* (Montello, 1993). To do this it accesses a *digital surface model*, sourced from LiDAR, which is a 2.5D representation of the city including buildings, vegetation, and land surface elevation. The Visibility Engine uses this dataset to offer a number of services, such as determining the line of sight from the observer to nominated points (e.g. which junctions are visible), and determining which entities within the city model are visible. Using these services, the IM determines if the destination is visible or not.

### 3.4 Question-Answering server

The QA server currently answers a range of *definition* and *biographical* questions such as, “Tell me more about the Scottish Parliament”, “Who was David Hume?”, “What is haggis?”, and requests to resume (eg. “Tell me more”). QA is also capable of recognizing out of scope requests, that is, either navigation-related questions that can be answered by computations from the City Model and dealt with elsewhere in the system (“How far away is the Scottish Parliament?”, “How do I get there?”), or exploration queries that cannot be handled yet (“When is the cannon gun fired from the castle?”). Question classification is entirely machine learning-based using the SMO algorithm (Keerthi et al., 1999) trained over 2013 annotated utterances. Once the question has been typed, QA proceeds to focus detection also using machine learning techniques (Mikhailian et al., 2009). Detected foci include possibly anaphoric expressions (“Who was he?”, “Tell me more about the castle”). These expressions are resolved against the dialogue history and geographical context. QA then proceeds to a textual search on texts from the Gazetteer of Scotland (Gittings, 2012) and Wikipedia, and definitions from WordNet glosses. The task is similar to TAC KBP 2013 Entity Linking Track and named en-

tity disambiguation (Cucerzan, 2007). Candidate answers are reranked using a trained confidence score with the top candidate used as the final answer. These are usually long, descriptive answers and are provided as a flow of sentence chunks that the user can interrupt (see table 2). The Interaction Manager queries the QA model and pushes information when a salient PoI is in the vicinity of the user.

<p><i>“Edinburgh’s most famous and historic thoroughfare, which has formed the heart of the Old Town since mediaeval times. The Royal Mile includes Castlehill, the Lawnmarket, the Canongate and the Abbey Strand, but, is officially known simply as the High Street.”</i></p>
--

Table 2: QA output: query on “Royal Mile”

### 3.5 Mobile client

The mobile client app, installed on an Android smartphone (Samsung Galaxy S3), connects the user to the dialogue system using a 3G data connection. The client senses the user’s location using positioning technology using GNSS satellites (GPS and GLONASS) which is sent to the dialogue system at the rate of one update every two seconds. It also sends pace rate of the user from the accelerometer sensor. In parallel, the client also places a phone call using which the user communicates with the dialogue system.

## 4 Baseline system

The baseline system chosen for evaluation was Samsung S-Voice, a state-of-the-art commercial smartphone speech interface. S-Voice is a Samsung Android mobile phone app that allows a user to use the functionalities of device using a speech interface. For example, the user can say “Call John” and it will dial John from the user’s contacts. It launches the Google Navigation app when users request directions and it activates Google Search for open ended touristic information questions. The Navigation app is capable of providing instructions in-situ using speech. We used the S-Voice system for comparison because it provided an integrated state-of-the-art interface to use both a navigation app and also an information-seeking app using the same speech interface. Users were encouraged to use these apps using speech but were allowed to use the GUI interface when using speech wasn’t working (e.g. misrecognition of local names). Users obtained the same kind of in-

formation (i.e. navigation directions, descriptions about entities such as people, places, etc) from the baseline system as they would from our system. However, our system interacted with the user using the speech modality only.

## 5 Experimental design

Spacebook and the baseline were evaluated in the summer of 2012. We evaluated both systems with 17 subjects in the streets of Edinburgh. There were 11 young subjects (between 20 and 26 years, mean=22 ± 2) and 6 older subjects (between 50 and 71 years, mean=61 ± 11). They were mostly native English speakers (88%). 59% of the users were regular smartphone users and their mean overall time spent in the city was 76 months. The test subjects had no previous experience with the proposed system. They were recruited via email adverts and mail shots. Subjects were given a task sheet with 8 tasks in two legs (4 tasks per leg). These tasks included both navigation and tourist information tasks (see table 3). Subjects used our system for one of the legs and the baseline system for the other and the order was balanced. Each leg took up to 30 mins to finish and the total duration including questionnaires was about 1.5 hours. Figure 2 shows the route taken by the subjects. The route is about 1.3 miles long. Subjects were followed by the evaluator who made notes on their behaviour (e.g. subject looks confused, subject looks at or manipulates the phone, subject looks around, etc).

Subjects filled in a demographic questionnaire prior to the experiment. After each leg, they filled in a system questionnaire (see appendix) rating their experience. After the end of the experiment, they filled out a comparative questionnaire and were debriefed. They were optionally asked to elaborate on their questionnaire ratings. Users were paid £20 after the experiment was over.

## 6 Results

Subjects were asked to identify tasks that they thought were successfully completed. The perceived task success rates of the two systems were compared for each task using the Chi square test. The results show that there is no statistically significant difference between the two systems in terms of perceived task success although the baseline system had a better task completion rate in tasks 1-3, 5 and 6. Our system performed better in

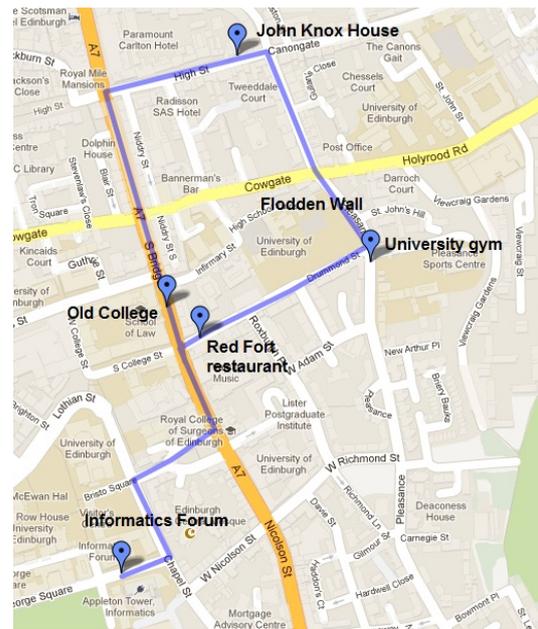


Figure 2: Task route

tourist information tasks (4, 7) (see table 4).

Task	Our system	Baseline	p
T1 (N)	77.7	100	0.5058
T2 (TI)	88.8	100	0.9516
T3 (N)	100	100	NA
T4 (TI)	100	87.5	0.9516
T5 (N+TI)	62.5	100	0.1654
T6 (N+TI)	87.5	100	0.9516
T7 (TI)	100	55.5	0.2926
T8 (N)	75.0	88.8	0.9105

Table 4: % Perceived Task success - task wise comparison (N - navigation task, TI - Tourist Information task)

The system questionnaires that were filled out by users after each leg were analysed. These consisted of questions concerning each system to be rated on a six point Likert scale (1-Strongly Disagree, 2-Disagree, 3-Somewhat Disagree, 4-Somewhat Agree, 5-Agree, 6-Strongly Agree). The responses were paired and tested using a Wilcoxon Sign Rank test. Median and Mode for each system and significance in differences are shown in table 5. Results show that although our system is not performing significantly better than the baseline system (SQ1-SQ10 except SQ7), users seem to find it more understanding (SQ7) and more interesting to interact with (SQ11) than the baseline. We grouped the subjects by age group and tested their responses. We found that the young subjects (age group 20-26), also felt that

<p>Leg 1</p> <p>(Task 1) Ask the system to guide you to the Red Fort restaurant.</p> <p>(Task 2) You've heard that Mary Queen of Scots lived in Edinburgh. Find out about her.</p> <p>(Task 3) Walk to the university gym.</p> <p>(Task 4) Near the gym there is an ancient wall with a sign saying "Flodden Wall". Find out what that is.</p> <p>Leg 2</p> <p>(Task 5) Try to find John Knox House and learn about the man.</p> <p>(Task 6) Ask the system to guide you to the Old College. What can you learn about this building?</p> <p>(Task 7) Try to find out more about famous Edinburgh people and places, for example, David Hume, John Napier, and Ian Rankin. Try to find information about people and places that you are personally interested in or that are related to what you see around you.</p> <p>(Task 8) Ask the system to guide you back to the Informatics Forum.</p>
--

Table 3: Tasks for the user

they learned something new about the city using it (SQ12) ( $p < 0.05$ ) while the elderly (age group 50-71) didn't. We also found statistically significant differences in smartphone users rating for our system on their learning compared to the baseline (SQ12).

Subjects were also asked to choose between the two systems given a number of requirements such as ease of use, use for navigation, tourist information, etc. There was an option to rank the systems equally (i.e. a tie). They were presented with the same requirements as the system questionnaire with one additional question - "Overall which system do you prefer?" (CQ0). Users' choice of system based on a variety of requirements is shown in table 6. Users' choice counts were tested using Chi-square test. Significant differences were found in users' choice of system for navigation and tourist information requirements. Users preferred the baseline system for navigation (CQ2) and our system for touristic information (CQ3) on the city. Although there was a clear choice of systems based on the two tasks, there was no significant preference of one system over the other overall (CQ0). They chose our system as the most interesting system to interact with (CQ11) and that it was more informative than the baseline (CQ12). Figure 3 shows the relative frequency between user choices on comparative questions. Figure 3 shows the relative frequency between user choices on comparative questions.

## 7 Analysis

Users found it somewhat difficult to navigate using Spacebook (see comments in table 7). Although the perceived task success shows that our system was able to get the users to their destination and there was no significant difference between the two systems based on their questionnaire response on navigation, they pointed out a number of issues and suggested a number of modifications. Many

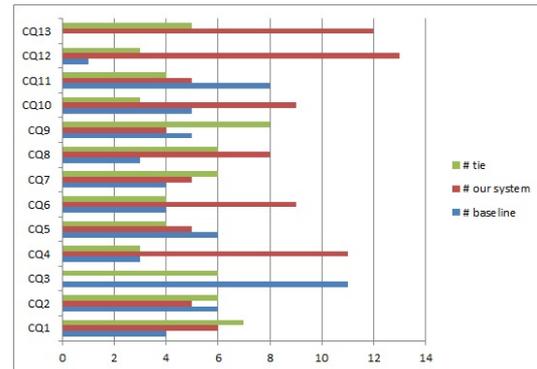


Figure 3: Responses to comparative questions

users noted that a visual map and the directional arrow in the baseline system was helpful for navigation. In addition, they noted that our system's navigation instructions were sometimes not satisfactory. They observed that there weren't enough instructions coming from the system at street junctions. They needed more confirmatory utterances (that they are walking in the right direction) (5 users) and quicker recovery and notification when walking the wrong way (5 users). They observed that the use of street names was confusing sometimes. Some users also wanted a route summary before the navigation instructions are given.

The problem with Spacebook's navigation policy was that it did not, for example, direct the user via easily visible landmarks (e.g. "Head towards the Castle"), and relies too much on street names. Also, due to the latency in receiving GPS information, the IM sometimes did not present instructions soon enough during evaluation. Sometimes it received erroneous GPS information and therefore got the user's orientation wrong. These problems will be addressed in the future version. Some users did find navigation instructions useful because of the use of proximal landmarks such

Question	B Mode	B Median	S Mode	S Median	p
SQ1 - Ease of use	4	4	5	4	0.8207
SQ2 - Navigation	4	4	5	4	0.9039
SQ3 - Tourist Information	2	3	4	4	0.07323
SQ4 - Easy to understand	5	5	5	5	0.7201
SQ5 - Useful messages	5	4	5	4	1
SQ6 - Response time	5	5	2	2	0.2283
SQ7 - Understanding	3	3	<b>5</b>	<b>4</b>	<b>0.02546</b>
SQ8 - Repetitive	2	3	2	3	0.3205
SQ9 - Aware of user environment	5	5	4	4	0.9745
SQ10 - Cues for guidance	5	5	5	5	0.1371
SQ11 - Interesting to interact with	5	4	<b>5</b>	<b>5</b>	<b>0.01799</b>
SQ12 - Learned something new	5	4	5	5	0.08942

Table 5: System questionnaire responses (B=Baseline, S=our system)

Task	Baseline Preferred	Our system Preferred	Tie	p-value
CQ0	23.52	35.29	41.17	0.66
CQ1	35.29	29.41	35.29	0.9429
CQ2	<b>64.70</b>	0	35.29	<b>0.004</b>
CQ3	17.64	<b>64.70</b>	17.64	<b>0.0232</b>
CQ4	35.29	29.41	23.52	0.8187
CQ5	23.52	52.94	23.52	0.2298
CQ6	23.52	29.41	35.29	0.8187
CQ7	17.64	47.05	35.29	0.327
CQ8	29.41	23.52	47.05	0.4655
CQ9	29.41	52.94	17.64	0.1926
CQ10	47.05	29.41	23.52	0.4655
CQ11	5.88	<b>76.47</b>	17.64	<b>0.0006</b>
CQ12	0	<b>70.58</b>	29.41	<b>0.005</b>

Table 6: User's choice on comparative questions (CQ are the same questions as SQ but requesting a ranking of the 2 systems)

as KFC, Tesco, etc. (popular chain stores). Some users also suggested that our system should have a map and that routes taken should be plotted on them for reference. Based on the ratings and observations made by the users, we conclude that our first hypothesis that Spacebook would be more efficient for navigation than the baseline because of its speech-only interface was inconclusive. We believe so because users' poor ratings for Spacebook may be due to the current choice of dialogue policy for navigation. It may be possible to reassure the user with a better dialogue policy with just the speech interface. However, this needs further investigation.

Users found the information-search task interesting and informative when they used Spacebook (see sample user comments in table 8). They also found push information on nearby PoIs unexpected and interesting as they would not have found them otherwise. Many users believed that this could be an interesting feature that could help tourists. They also found that asking questions and

finding answers was much easier with Spacebook compared to the baseline system, where sometimes users needed to type search keywords in. Another user observation was that they did not have to stop to listen to information presented by our system (as it was in speech) and could carry on walking. However, with the baseline system, they had to stop to read information off the screen. Although users in general liked the QA feature, many complained that Spacebook spoke too quickly when it was presenting answers. Some users felt that the system might lose context of the navigation task if presented with a PoI question. In contrast, some others noted Spacebook's ability to interleave the two tasks and found it to be an advantage.

Users' enthusiasm for our system was observed when (apart from the points of interest that were in the experimental task list) they also asked spontaneous questions about James Watt, the Talbot Rice gallery, the Scottish Parliament and Edinburgh Castle. Some of the PoIs that the system pushed information about were the Royal College of Surgeons, the Flodden Wall, the Museum of Childhood, and the Scottish Storytelling Centre. Our system answered a mean of 2.5 out of 6.55 questions asked by users in leg 1 and 4.88 out of 8.5 questions in leg 2. Please note that an utterance is sent to QA if it is not parsed by the parser and therefore some utterances may not be legitimate questions themselves. Users were pushed a mean of 2.88 and 6.37 PoIs during legs 1 and 2. There were a total of 17 "tell me more" requests requesting the system to present more information (mean=1.35 ± 1.57).

Evaluators who followed the subjects noted that the subjects felt difficulty using the baseline system as they sometimes struggled to see the screen

1. "It's useful when it says 'Keep walking' but it should say it more often."
2. "[Your system] not having a map, it was sometimes difficult to check how aware it was of my environment."
3. "[Google] seemed to be easier to follow as you have a map as well to help."
4. "It told me I had the bank and Kentucky Fried Chicken so I crossed the road because I knew it'd be somewhere over beside them. I thought 'OK, great. I'm going the right way,' but then it didn't say anything else. I like those kind of directions because when it said to go down Nicolson Street I was looking around trying to find a street sign."
5. "The system keeps saying 'when we come to a junction, I will tell you where to go', but I passed junctions and it didn't say anything. It should say 'when you need to change direction, I will tell you.'"
6. "I had to stop most of the times for the system to be aware of my position. If walking very slowly, its awareness of both landmarks and streets is excellent."

Table 7: Sample user comments on the navigation task

1. "Google doesn't \*offer\* any information. I would have to know what to ask for..."
2. "Since many information is given without being asked for (by your system), one can discover new places and landmarks even if he lives in the city. Great feature!!"
3. "I didn't feel confident to ask [your system] a question and still feel it would remember my directions"
4. "Google could only do one thing at a time, you couldn't find directions for a place whilst learning more."
5. "If she talked a little bit slower [I would use the system for touristic purposes]. She just throws masses of information really, really quickly."

Table 8: Sample user comments on the tourist information task

in bright sunlight. They sometimes had difficulty identifying which way to go based on the route plotted on the map. In comparison, subjects did not have to look at the screen when they used our system. Based on the ratings and observations made by the users about our system's tourist information features such as answering questions and pushing PoI information, we have support for our second hypothesis: that users find a dialogue interface which integrates question-answering and navigation within a shared context to be useful for finding information about entities in the urban environment.

## 8 Future plans

We plan to extend Spacebook's capabilities to address other challenges in pedestrian navigation and tourist information. Many studies have shown that visible landmarks provide better cues for navigation than street names (Ashweeni and Steed, 2006; Hiley et al., 2008). We will use visible landmarks identified using the visibility engine to make navigation instructions more effective, and we plan to include entities in dialogue and visual context as candidates for PoI push, and to implement an adaptive strategy that will estimate user interests and push information that is of interest to them. We are also taking advantage of user's local knowledge of the city to present navigation instructions only for the part of the route that the user does not have any knowledge of. These features, we believe, will make users' experience of

the interface more pleasant, useful and informative.

## 9 Conclusion

We presented a mobile dialogue app called Spacebook to support pedestrian users in navigation and tourist information gathering in urban environments. The system is a speech-only interface and addresses navigation and tourist information in an integrated way, using a shared dialogue context. For example, using the navigational context, Spacebook can push point-of-interest information which can then initiate touristic exploration tasks using the QA module.

We evaluated the system against a state-of-the-art baseline (Samsung S-Voice with Google Navigation and Search) with a group of 17 users in the streets of Edinburgh. We found that users found Spacebook interesting to interact with, and that it was their system of choice for touristic information exploration tasks. These results were statistically significant. Based on observations and user ratings, we conclude that our speech-only system was less preferred for navigation and more preferred for tourist information tasks due to features such as PoI pushing and the integrated QA module, when compared to the baseline system. Younger users, who used Spacebook, even felt that they learned new facts about the city.

## Acknowledgments

The research leading to these results was funded by the European Commission's Framework 7 programme under grant

agreement no. 270019 (SPACEBOOK project).

## References

- K. B. Ashweeni and A. Steed. 2006. A natural wayfinding exploiting photos in pedestrian navigation systems. In *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*.
- P. Bartie and W. Mackaness. 2006. Development of a speech-based augmented reality system to support exploration of cityscape. *Transactions in GIS*, 10:63–86.
- P. Bartie and W. Mackaness. 2012. D3.4 Pedestrian Position Tracker. Technical report, The SPACEBOOK Project (FP7/2011-2014 grant agreement no. 270019).
- P. Bartie and W. Mackaness. 2013. D3.1.2 The SpaceBook City Model. Technical report, The SPACEBOOK Project (FP7/2011-2014 grant agreement no. 270019).
- D. Byron, A. Koller, J. Oberlander, L. Stoia, and K. Striegnitz. 2007. Generating Instructions in Virtual Environments (GIVE): A challenge and evaluation testbed for NLG. In *Proceedings of the Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation*.
- S. Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of EMNLP-CoNLL*.
- R. Dale, S. Geldof, and J. Prost. 2003. CORAL : Using Natural Language Generation for Navigational Assistance. In *Proceedings of ACSC2003, Australia*.
- Nina Dethlefs and Heriberto Cuayáhuitl. 2011. Hierarchical Reinforcement Learning and Hidden Markov Models for Task-Oriented Natural Language Generation. In *Proc. of ACL*.
- B. Gittings. 2012. The Gazetteer for Scotland - <http://www.scottish-places.info>.
- H. Hiley, R. Vedantham, G. Cuellar, A. Liuy, N. Gelfand, R. Grzeszczuk, and G. Borriello. 2008. Landmark-based pedestrian navigation from collections of geotagged photos. In *Proceedings of the 7th Int. Conf. on Mobile and Ubiquitous Multimedia (MUM)*.
- S. Janarthanam and O. Lemon. 2011. The GRUVE Challenge: Generating Routes under Uncertainty in Virtual Environments. In *Proceedings of ENLG*.
- S. Janarthanam, O. Lemon, X. Liu, P. Bartie, W. Mackaness, T. Dalmas, and J. Goetze. 2012. Integrating location, visibility, and Question-Answering in a spoken dialogue system for Pedestrian City Exploration. In *Proc. of SIGDIAL 2012, S. Korea*.
- H. Kashioka, T. Misu, E. Mizukami, Y. Shiga, K. Kayama, C. Hori, and H. Kawai. 2011. Multimodal Dialog System for Kyoto Sightseeing Guide. In *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*.
- S.S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. 1999. Improvements to Platt’s SMO Algorithm for SVM Classifier Design. *Neural Computation*, 3:637–649.
- J. Ko, F. Murase, T. Mitamura, E. Nyberg, M. Tateishi, I. Akahori, and N. Hataoka. 2005. CAMMIA: A Context-Aware Spoken Dialog System for Mobile Environments. In *IEEE ASRU Workshop*.
- C. Kray, K. Laakso, C. Elting, and V. Coors. 2003. Presenting Route Instructions on Mobile Devices. In *Proceedings of IUI 03, Florida*.
- R. Malaka and A. Zipf. 2000. Deep Map - challenging IT research in the framework of a tourist information system. In *Information and Communication Technologies in Tourism 2000*, pages 15–27. Springer.
- A. Mikhailsian, T. Dalmas, and R. Pinchuk. 2009. Learning foci for question answering over topic maps. In *Proceedings of ACL 2009*.
- D. Montello. 1993. Scale and multiple psychologies of space. In A. U. Frank and I. Campari, editors, *Spatial information theory: A theoretical basis for GIS*.
- M. Raubal and S. Winter. 2002. Enriching wayfinding instructions with local landmarks. In *Second International Conference GIScience. Springer, USA*.
- C.J. Shroder, W. Mackaness, and B. Gittings. 2011. Giving the Right Route Directions: The Requirements for Pedestrian Navigation Systems. *Transactions in GIS*, pages 419–438.
- N. Webb and B. Webber. 2009. Special Issue on Interactive Question Answering: Introduction. *Natural Language Engineering*, 15(1):1–8.
- P. A. Zandbergen and S. J. Barbeau. 2011. Positional Accuracy of Assisted GPS Data from High-Sensitivity GPS-enabled Mobile Phones. *Journal of Navigation*, 64(3):381–399.

# Lightly Supervised Learning of Procedural Dialog Systems

**Svitlana Volkova**

CLSP

Johns Hopkins University

Baltimore, MD

svitlana@jhu.edu

**Pallavi Choudhury, Chris Quirk, Bill Dolan**

NLP Group

Microsoft Research

Redmond, WA

pallavic, chrisq,

billdol@microsoft.com

**Luke Zettlemoyer**

Computer Science and Engineering

University of Washington

Seattle, WA

lsz@cs.washington.edu

## Abstract

Procedural dialog systems can help users achieve a wide range of goals. However, such systems are challenging to build, currently requiring manual engineering of substantial domain-specific task knowledge and dialog management strategies. In this paper, we demonstrate that it is possible to learn procedural dialog systems given only light supervision, of the type that can be provided by non-experts. We consider domains where the required task knowledge exists in textual form (*e.g.*, instructional web pages) and where system builders have access to statements of user intent (*e.g.*, search query logs or dialog interactions). To learn from such textual resources, we describe a novel approach that first automatically extracts task knowledge from instructions, then learns a dialog manager over this task knowledge to provide assistance. Evaluation in a Microsoft Office domain shows that the individual components are highly accurate and can be integrated into a dialog system that provides effective help to users.

## 1 Introduction

Procedural dialog systems aim to assist users with a wide range of goals. For example, they can guide visitors through a museum (Traum et al., 2012; Aggarwal et al., 2012), teach students physics (Steinhauser et al., 2011; Dzikovska et al., 2011), or enable interaction with a health care

U: "I want to add page numbers and a title"
S: "Top or Bottom of the page?"
U: "Top"
S: "Please select page design from the templates" (*System shows drop down menu*)
U: *User selects from menu*
S: "Enter header or footer content"
U: "C.V."
S: "Task completed."

Figure 1: An example dialog interaction between a system (S) and user (U) that can be automatically achieved by learning from instructional web page and query click logs.

system (Morbini et al., 2012; Rizzo et al., 2011). However, such systems are challenging to build, currently requiring expensive, expert engineering of significant domain-specific task knowledge and dialog management strategies.

In this paper, we present a new approach for learning procedural dialog systems from task-oriented textual resources in combination with light, non-expert supervision. Specifically, we assume access to task knowledge in textual form (*e.g.*, instructional web pages) and examples of user intent statements (*e.g.*, search query logs or dialog interactions). Such instructional resources are available in many domains, ranging from recipes that describe how to cook meals to software help web pages that describe how to achieve goals by interacting with a user interface.<sup>1</sup>

<sup>1</sup>ehow.com, wikianswers.com

There are two key challenges: we must (1) learn to convert the textual knowledge into a usable form and (2) learn a dialog manager that provides robust assistance given such knowledge. For example, Figure 1 shows the type of task assistance that we are targeting in the Microsoft Office setting, where the system should learn from web pages and search query logs. Our central contribution is to show that such systems can be built without the help of knowledge engineers or domain experts. We present new approaches for both of our core problems. First, we introduce a method for learning to map instructions to tree representations of the procedures they describe. Nodes in the tree represent points of interaction with the questions the system can ask the user, while edges represent user responses. Next, we present an approach that uses example user intent statements to simulate dialog interactions, and learns how to best map user utterances to nodes in these induced dialog trees. When combined, these approaches produce a complete dialog system that can engage in conversations by automatically moving between the nodes of a large collection of induced dialog trees.

Experiments in the Windows Office help domain demonstrate that it is possible to build an effective end-to-end dialog system. We evaluate the dialog tree construction and dialog management components in isolation, demonstrating high accuracy (in the 80-90% range). We also conduct a small-scale user study which demonstrates that users can interact productively with the system, successfully completing over 80% of their tasks. Even when the system does fail, it often does so in a graceful way, for example by asking redundant questions but still reaching the goal within a few additional turns.

## 2 Overview of Approach

Our task-oriented dialog system understands user utterances by mapping them to nodes in dialog trees generated from instructional text. Figure 2 shows an example of a set of instructions and the corresponding dialog tree. This section describes the problems that we must solve to enable such interactions, and outlines our approach for each.

**Knowledge Acquisition** We extract task knowledge from instructional text (*e.g.*, Figure 2, left) that describes (1) actions to be performed, such as clicking a button, and (2) places where input is needed from the user, for example to enter the

contents of the footer or header they are trying to create. We aim to convert this text into a form that will enable a dialog system to automatically assist with the described task. To this end, we construct *dialog trees* (*e.g.*, Figure 2, right) with nodes to represent entire documents (labeled as topics  $t$ ), nodes to represent user goals or intents ( $g$ ), and system action nodes ( $a$ ) that enable execution of specific commands. Finally, each node has an associated system action  $a_s$ , which can prompt user input (*e.g.*, with the question “Top or bottom of the page?”) and one or more user actions  $a_u$  that represent possible responses. All nodes connect to form a tree structure that follows the workflow described in the document. Section 3 presents a scalable approach for inducing dialog trees.

**Dialog Management** To understand user intent and provide task assistance, we need a dialog management approach that specifies what the system should do and say. We adopt a simple approach that at all times maintains an index into a node in a dialog tree. Each system utterance is then simply the action  $a_s$  for that node. However, the key challenge comes in interpreting user utterances. After each user statement, we must automatically update our node index. At any point, the user can state a general goal (*e.g.*, “I want to add page numbers”), refine their goal (*e.g.*, “in a footer”), or both (*e.g.*, “I want to add page numbers in the footer”). Users can also change their goals in the process of completing the tasks.

We develop a simple classification approach that is robust to these different types of user behavior. Specifically, we learn classifiers that, given the dialog interaction history, predict how to pick the next tree node from the space of all nodes in the dialog trees that define the task knowledge. We isolate two specific cases, classifying initial user utterances (Section 4) and classifying all subsequent utterances (Section 5). This approach allows us to isolate the difference in language for the two cases, and bias the second case to prefer tree nodes near the current one. The resulting approach allows for significant flexibility in traversing the dialog trees.

**Data and Evaluation** We collected a large set of such naturally-occurring web search queries that resulted in a user click on a URL in the Microsoft Office help domain.<sup>2</sup> We found that queries longer than 4-5 words often resembled natural language utterances that could be used for dialog interac-

<sup>2</sup><http://office.microsoft.com>

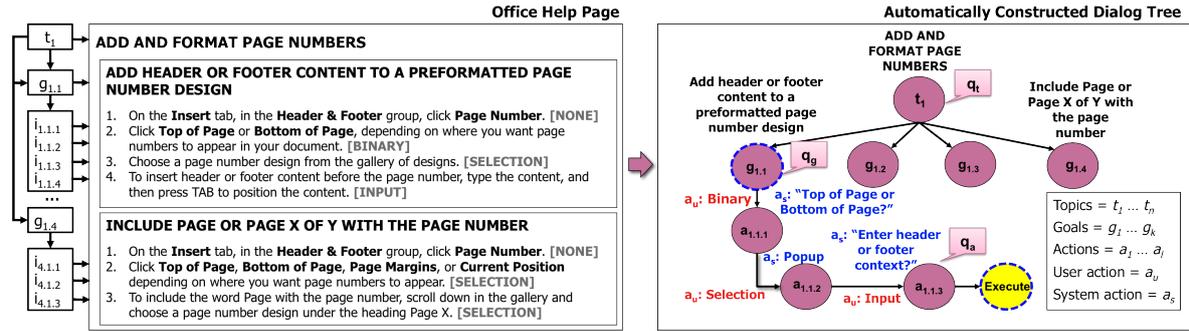


Figure 2: An example instructional text paired with a section of the corresponding dialog tree.

tions, for example *how do you add borders*, *how can I add a footer*, *how to insert continuous page numbers*, and *where is the header and footer*.

We also collected instructional texts from the web pages that describe how to solve 76 of the most pressing user goals, as indicated by query click log statistics. On average 1,000 user queries were associated with each goal. To some extent clickthroughs can be treated as a proxy for user frustration; popular search targets probably represent user pain points.

### 3 Building Dialog Trees from Instructions

Our first problem is to convert sets of instructions for user goals to dialog trees, as shown in Figure 2. These goals are broadly grouped into topics (instructional pages). In addition, we manually associate each node in a dialog tree with a training set of 10 queries. For the 76 goals (246 instructions) in our data, this annotation effort took a single annotator a total of 41 hours. Scaling this approach to the entire Office help domain would require a focused annotation effort. Crucially, though, this annotation work can be carried out by non-specialists, and could even be crowdsourced (Bernstein et al., 2010).

**Problem Definition** As input, we are given instructional text ( $p_1 \dots p_n$ ), comprised of topics ( $t_1 \dots t_n$ ) describing:

- (1) high-level user intents (e.g.,  $t_1$  – “add and format page numbers”)
- (2) goals ( $g_1, \dots, g_k$ ) that represent more specific user intents (e.g.,  $g_1$  – “add header or footer content to a preformatted page number design”,  $g_2$  – “place the page number in the side margin of the page”).

Given instructional text  $p_1 \dots p_n$  and queries  $q_1 \dots q_m$  per topic  $t_i$ , our goals are as follows:

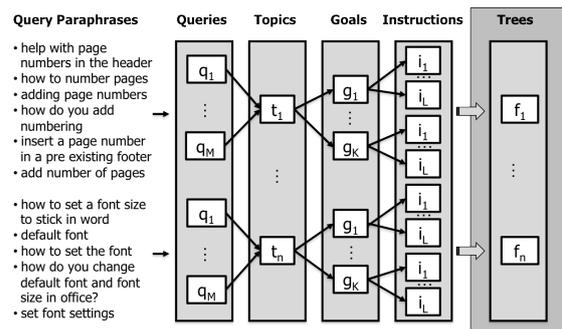


Figure 3: Relationships between user queries and OHP with goals, instructions and dialog trees.

- for every instructional page  $p_i$  extract a topic  $t_i$  and a set of goals  $g_1 \dots g_k$ ;
- for every goal  $g_j$  for a topic  $t_i$ , extract a set of instructions  $i_1 \dots i_l$ ;
- from topics, goals and instructions, construct dialog trees  $f_1 \dots f_n$  (one dialog tree per topic). Classify instructions to user interaction types thereby identifying system action nodes  $a_s^1 \dots a_s^l$ . Transitions between these nodes are the user actions  $a_u^1 \dots a_u^l$ .

Figure 2 (left) presents an example of a topic extracted from the help page, and a set of goals and instructions annotated with user action types.

In the next few sections of the paper, we outline an overall system component design demonstrating how queries and topics are mapped to the dialog trees in Figure 3. The figure shows *many-to-one* relations between queries and topics, *one-to-many* relations between topics and goals, goals and instructions, and *one-to-one* relations between topics and dialog trees.

**User Action Classification** We aim to classify instructional text ( $i_1 \dots i_l$ ) for every goal  $g_j$  in the decision tree into four categories: *binary*, *selection*, *input* or *none*.

Given a single instruction  $i$  with category  $a_u$ , we use a log-linear model to represent the distri-

bution over the space of possible user actions. Under this representation, the user action distribution is defined as:

$$p(a_u|i, \theta) = \frac{e^{\theta \cdot \phi(a_u, i)}}{\sum_{a'_u} e^{\theta \cdot \phi(a'_u, i)}}, \quad (1)$$

where  $\phi(a_u, i) \in \mathbb{R}^n$  is an  $n$ -dimensional feature representation and  $\vec{\theta}$  is a parameter vector we aim to learn. Features are indicator functions of properties of the instructions and a particular class. For smoothing we use a zero mean, unit variance Gaussian prior  $(0, 1)$  that penalizes  $\vec{\theta}$  for drifting too far from the mean, along with the following optimization function:

$$\begin{aligned} \log p(A_u, \theta|I) &= \log p(A_u|I, \theta) - \log p(\theta) = \\ &= \sum_{a_u, i \in (A_u, I)} p(a_u|i, \theta) - \sum_i \frac{(\theta - \mu_i)^2}{2\sigma_i^2} + k \end{aligned} \quad (2)$$

We use L-BFGS (Nocedal and Wright, 2000) as an optimizer.

**Experimental Setup** As described in Section 2, our dataset consists of 76 goals grouped into 30 topics (average 2-3 goals per topic) for a total of 246 instructions (average 3 instructions per goal). We manually label all instructions with user action  $a_u$  categories. The distribution over categories is *binary=14*, *input=23*, *selection=80* and *none=129*. The data is skewed towards the categories *none* and *selection*. Many instruction do not require any user input and can be done automatically, e.g., “On the Insert tab, in the Header and Footer group, click Page Number”. The example instructions with corresponding user action labels are shown in Figure 2 (left). Finally, we divide the 246 instructions into 2 sets: 80% training and 20% test, 199 and 47 instructions respectively.

**Results** We apply the user action type classification model described in the Eq.1 and Eq.2 to classify instructions from the test set into 4 categories. In Table 1 we report classification results for 2 baselines: a majority class and heuristic-based approach, and 2 models with different feature types: *ngrams* and *ngrams + stems*. For a heuristic baseline, we use simple lexical clues to classify instructions (e.g., *X or Y* for *binary*, *select Y* for *selection* and *type X, insert Y* for *input*). Table 1 summarizes the results of mapping instructional text to user actions.

Features	# Features	Accuracy
Baseline 1: Majority	–	0.53
Baseline 2: Heuristic	–	0.64
Ngrams	10,556	0.89
Ngrams + Stems	12,196	0.89

Table 1: Instruction classification results.

**Building the Dialog Trees** Based on the classified user action types, we identify system actions  $a_s^1 \dots a_s^l$  which correspond to 3 types of user actions  $a_s^1 \dots a_s^l$  (excluding *none* type) for every goal in a topic  $t_i$ . This involved associating all words from an instruction  $i_l$  with a system action  $a_s^l$ . Finally, for every topic we automatically construct a dialog tree as shown in Figure 2 (right). The dialog tree includes a topic  $t_1$  with goals  $g_1 \dots g_4$ , and actions (user actions  $a_u$  and system actions  $a_s$ ).

**Definition 1.** A dialog tree encodes a user-system dialog flow about a topic  $t_i$  represented as a directed unweighted graph  $f_i = (V, E)$  where topics, goals and actions are nodes of corresponding types  $\{t_1 \dots t_n\}, \{g_1 \dots g_k\}, \{a_1 \dots a_l\} \in V$ . There is a hierarchical dependency between topic, goal and action nodes. User interactions are represented by edges  $t_i \rightarrow \{g_1 \dots g_k\}, a_u^1 = (g_j, a_1) \dots a_u^l = (a_{k-1}, a_k) \in E$ .

For example, in the dialog tree in Figure 2 there is a relation  $t_1 \rightarrow g_4$  between the topic  $t_1$  “add and format page numbers” and the goal  $g_4$  “include page of page X of Y with the page number”. Moreover, in the dialog tree, the topic level node has one index  $i \in [1..n]$ , where  $n$  is the number of topics. Every goal node includes information about its parent (topic) node and has double index  $i.j$ , where  $j \in [1..k]$ . Finally, action nodes include information about their parent (goal) and grandparent (topic) nodes and have triple index  $i.j.z$ , where  $z \in [1..l]$ .

## 4 Understanding Initial Queries

This section presents a model for classifying initial user queries to nodes in a dialog tree, which allows for a variety of different types of queries. They can be under-specified, including information about a topic only (e.g., “add or delete page numbers”); partially specified, including information about a goal (e.g., “insert page number”); or over-specified, including information about an action (e.g., “page numbering at bottom page”).

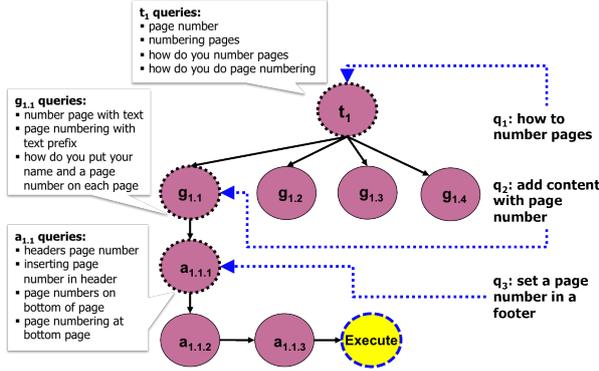


Figure 4: Mapping initial user queries to the nodes on different depth in a dialog tree.

**Problem Definition** Given an initial query, the dialog system initializes to a state  $s_0$ , searches for the deepest relevant node given a query, and maps the query to a node on a topic  $t_i$ , goal  $g_j$  or action  $a_k$  level in the dialog tree  $f_i$ , as shown in Figure 4.

More formally, as input, we are given automatically constructed dialog trees  $f_1 \dots f_n$  for instructional text (help pages) annotated with topic, goal and action nodes and associated with system actions as shown in Figure 2 (right). From the query logs, we associate queries with each node type: topic  $q_t$ , goal  $q_g$  and action  $q_a$ . This is shown in Figure 2 and 4. We join these dialog trees representing different topics into a *dialog network* by introducing a global root. Within the network, we aim to find (1) an initial dialog state  $s_0$  that maximizes the probability of state given a query  $p(s_0|q, \theta)$ ; and (2) the deepest relevant node  $v \in V$  on topic  $t_i$ , goal  $g_j$  or action  $a_k$  depth in the tree.

**Initial Dialog State Model** We aim to predict the best node in a dialog tree  $t_i, g_j, a_l \in V$  based on a user query  $q$ . A query-to-node mapping is encoded as an initial dialog state  $s_0$  represented by a binary vector over all nodes in the dialog network:

$$s_0 = [t_1, g_{1.1}, g_{1.2}, g_{1.2.1} \dots, t_n, g_{n.1}, g_{n.1.1}].$$

We employ a log-linear model and try to maximize initial dialog state distribution over the space of all nodes in a dialog network:

$$p(s_0|q, \theta) = \frac{e^{\sum_i \theta_i \phi_i(s_0, q)}}{\sum_{s'_0} e^{\sum_i \theta_i \phi_i(s'_0, q)}}, \quad (3)$$

Optimization follows Eq. 2.

We experimented with a variety of features. Lexical features included query *ngrams* (up to 3-grams) associated with every node in a dialog tree with removed stopwords and stemming query unigrams. We also used network structural features:

Features	Accuracy		
	Topic	Goal	Action
Random	0.10	0.04	0.04
TFIDF 1Best	0.81	0.21	0.45
Lexical (L)	0.92	0.66	0.63
L + 10TFIDF	0.94	0.66	0.64
L + 10TFIDF + PO	0.94	0.65	0.65
L + 10TFIDF + QO	0.95	0.72	0.69
All above + QHistO	0.96	0.73	0.71

Table 2: Initial dialog state classification results where L stands for lexical features, 10TFIDF - 10 best tf-idf scores, PO - prompt overlap, QO - query overlap, and QHistO - query history overlap.

tf-idf scores, query *ngram* overlap with the topic and goal descriptions, as well as system action prompts, and query *ngram* overlap with a history including queries from parent nodes.

**Experimental Setup** For each dialog tree, nodes corresponding to single instructions were hand-annotated with a small set of user queries, as described in Section 3. Approximately 60% of all action nodes have no associated queries<sup>3</sup> For the 76 goals, the resulting dataset consists of 972 node-query pairs, 80% training and 20% test.

**Results** The initial dialog state classification model of finding a single node given an initial query is described in Eq. 3.

We chose two simple baselines: (1) randomly select a node in a dialog network and (2) use a tf-idf 1-best model.<sup>4</sup> Stemming, stopword removal and including top 10 tf-idf results as features led to a 19% increase in accuracy on an action node level over baseline (2). Adding the following features led to an overall 26% improvement: query overlap with a system prompt (PO), query overlap with other node queries (QO), and query overlap with its parent queries (QHistO).

We present more detailed results for topic, goal and action nodes in Table 2. For nodes deeper in the network, the task of mapping a user query to an action becomes more challenging. Note, however, that the action node accuracy numbers actually un-

<sup>3</sup>There are multiple possible reasons for this: the software user interface may already make it clear how to accomplish this intent, the user may not understand that the software makes this fine-grained option available to them, or their experience with search engines may lead them to state their intent in a more coarse-grained way.

<sup>4</sup>We use cosine similarity to rank all nodes in a dialog network and select the node with the highest rank.

derstate the utility of the resulting dialog system. The reason is that even incorrect node assignments can lead to useful system performance. As long as a misclassification results being assigned to a too-high node within the correct dialog tree, the user will experience a graceful failure: they may be forced to answer some redundant questions, but they will still be able to accomplish the task.

## 5 Understanding Query Refinements

We also developed a classifier model for mapping followup queries to the nodes in a dialog network, while maintaining a dialog state that summarizes the history of the current interaction.

**Problem Definition** Similar to the problem definition in Section 4, we are given a network of dialog trees  $f_1 \dots f_n$  and a query  $q'$ , but in addition we are given the previous dialog state  $s$ , which contains the previous user utterance  $q$  and the last system action  $a_s$ . We aim to find a new dialog state  $s'$  that pairs a node from the dialog tree with updated history information, thereby undergoing a dialog state update.

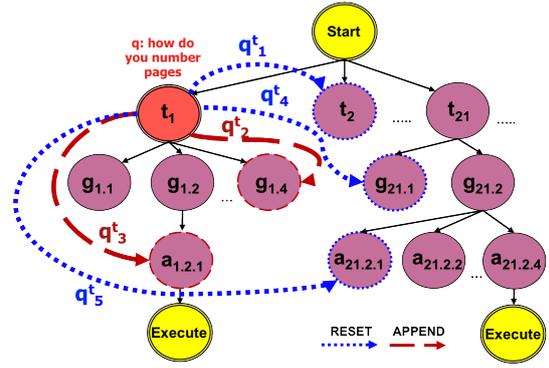
We learn a linear classifier that models  $p(s'|q', q, a_s, \theta)$ , the dialog state update distribution, where we constrain the new state  $s'$  to contain the new utterance  $q'$  we are interpreting. This distribution models 3 transition types: *append*, *override* and *reset*.

**Definition 2.** An *append* action defines a dialog state update when transitioning from a node to its children at any depth in the same dialog tree e.g.,  $t_i \rightarrow g_{i,j}$  (from a topic to a goal node),  $g_{i,j} \rightarrow a_{i,j,z}$  (from a goal to an action node) etc.

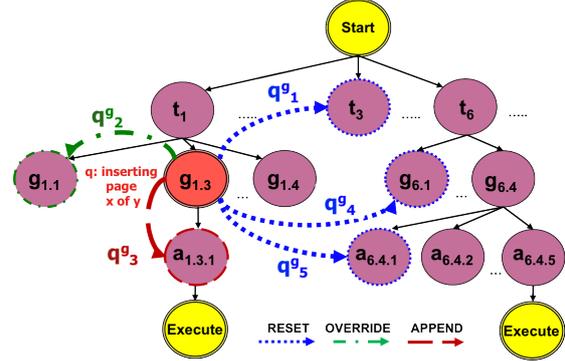
**Definition 3.** An *override* action defines a dialog state update when transitioning from a goal to its sibling node. It could also be from an action node<sup>5</sup> to another in its parent sibling node in the same dialog tree e.g.,  $g_{i,j-1} \rightarrow g_{i,j}$  (from one goal to another goal in the same topic tree),  $a_{i,j,z} \rightarrow a_{i,-j,z}$  (from an action node to another action node in a different goal in the same dialog tree) etc.

**Definition 4.** A *reset* action defines a dialog state update when transitioning from a node in a current dialog tree to any other node at any depth in a dialog tree other than the current dialog tree e.g.,  $t_i \rightarrow t_{-i}$ , (from one topic node to another topic

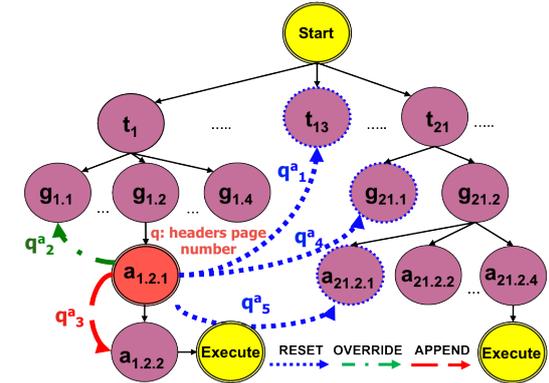
<sup>5</sup>A transition from  $a_{i,j,z}$  must be to a different goal or an action node in a different goal but in the same dialog tree.



(a) Updates from topic node  $t_i$



(b) Updates from goal node  $g_j$



(c) Updates from action node  $a_i$

Figure 5: Information state updates: append, reset and override updates based on Definition 2, 3 and 4, respectively, from topic, goal and action nodes.

node)  $t_i \rightarrow g_{-i,j}$  (from a topic node to a goal node in a different topic subtree), etc.

The *append* action should be selected when the user’s intent is to clarify a previous query (e.g., “insert page numbers”  $\rightarrow$  “page numbers in the footer”). An *override* action is appropriate when the user’s intent is to change a goal within the same topic (e.g., “insert page number  $\rightarrow$  “change page number”). Finally, a *reset* action should be used when the user’s intent is to restart the dialog (e.g., “insert page x of y”  $\rightarrow$  “set default font”). We present more examples for *append*, *override* and *reset* dialog state update actions in Table 3.

Previous Utterance, $q$	User Utterance, $q'$	Transition	Update Action, $a$
inserting page numbers	$q_1^t$ add a background	$t_i \rightarrow t_{-i}$	2, reset-T, reset
how to number pages	$q_2^t$ insert numbers on pages in margin	$t_i \rightarrow s_{i,j}$	1.4, append-G, append
page numbers	$q_3^t$ set a page number in a footer	$t_i \rightarrow a_{i,j,z}$	1.2.1, append-A, append
page number a document	$q_4^t$ insert a comment	$t_i \rightarrow g_{-i,j}$	21.1, reset-G, reset
page number	$q_5^t$ add a comment "redo"	$t_i \rightarrow a_{-i,j,z}$	21.2.1, reset-A, reset
page x of y	$q_1^g$ add a border	$g_{i,j} \rightarrow t_{-i}$	6, reset-T, reset
format page x of x	$q_2^g$ enter text and page numbers	$g_{i,j} \rightarrow g_{i,-j}$	1.1, override-G, override
enter page x of y	$q_3^g$ page x of y in footer	$g_{i,j} \rightarrow a_{i,j,z}$	1.3.1, append-A, append
inserting page x of y	$q_4^g$ setting a default font	$g_{i,j} \rightarrow g_{-i,j}$	6.1, reset-G, reset
showing page x of x	$q_5^g$ set default font and style	$g_{i,j} \rightarrow a_{-i,j,z}$	6.4.1, reset-A, reset
page numbers bottom	$q_1^a$ make a degree symbol	$a_{i,j,z} \rightarrow t_{-i}$	13, reset-T, reset
numbering at bottom page	$q_2^a$ insert page numbers	$a_{i,j,z} \rightarrow g_{i,-j}$	1.1, override-G, override
insert footer page numbers	$q_3^a$ page number design	$a_{i,j,z-1} \rightarrow a_{i,j,z}$	1.2.2, append-A, append
headers page number	$q_4^a$ comments in document	$a_{i,j,z} \rightarrow g_{-i,j}$	21.1, reset-G, reset
page number in a footer	$q_5^a$ changing initials in a comment	$a_{i,j,z} \rightarrow a_{-i,j,z}$	21.2.1, reset-A, reset

Table 3: Example  $q$  and  $q'$  queries for append, override and reset dialog state updates.

Figure 5 illustrates examples of *append*, *override* and *reset* dialog state updates. All transitions presented in Figure 5 are aligned with the example  $q$  and  $q'$  queries in Table 3.

**Dialog State Update Model** We use a log-linear model to maximize a dialog state distribution over the space of all nodes in a dialog network:

$$p(s'|q', q, a_s, \theta) = \frac{e^{\sum_i \theta_i \phi_i(s', q', a_s, q)}}{\sum_{s''} e^{\sum_i \theta_i \phi_i(s'', q', a_s, q)}}, \quad (4)$$

Optimization is done as described in Section 3.

**Experimental Setup** Ideally, dialog systems should be evaluated relative to large volumes of real user interaction data. Our query log data, however, does not include dialog turns, and so we turn to simulated user behavior to test our system.

Our approach, inspired by recent work (Schatzmann et al., 2006; Scheffler and Young, 2002; Georgila et al., 2005), involves simulating dialog turns as follows. To define a state  $s$  we sample a query  $q$  from a set of queries per node  $v$  and get a corresponding system action  $a_s$  for this node; to define a state  $s'$ , we sample a new query  $q'$  from another node  $v' \in V, v \neq v'$  which is sampled using a prior probability biased towards *append*:  $p(\text{append})=0.7$ ,  $p(\text{override})=0.2$ ,  $p(\text{reset})=0.1$ . This prior distribution defines a dialog strategy where the user primarily continues the current goal and rarely resets.

We simulate 1100 previous state and new query pairs for training and 440 pairs for testing. The features were lexical, including word ngrams, stems with no stopwords; we also tested network structure, such as:

- old  $q$  and new  $q'$  query overlap (QO);
- $q'$  overlap with a system prompt  $a_s$  (PO);

- $q'$  ngram overlap with all queries from the old state  $s$  (SQO);
- $q'$  ngram overlap with all queries from the new state  $s'$  (S'QO);
- $q'$  ngram overlap with all queries from the new state parents (S'ParQO).

**Results** Table 4 reports results for dialog state updates for topic, goal and action nodes. We also report performance for two types of dialog updates such as: *append* (App.) and *override* (Over.).

We found that the combination of lexical and query overlap with the previous and new state queries yielded the best accuracies: 0.95, 0.84 and 0.83 for topic, goal and action node level, respectively. As in Section 4, the accuracy on the topic level node was highest. Perhaps surprisingly, the reset action was perfectly predicted (accuracy is 100% for all feature combinations, not included in figure). The accuracies for *append* and *override* actions are also high (*append* 95%, *override* 90%).

Features	Topic	Goal	Action	App.	Over.
L	0.92	0.76	0.78	0.90	0.89
L+Q	0.93	0.80	0.80	0.92	0.83
L+P	0.93	0.80	0.79	0.91	0.85
L+Q+P	0.94	0.80	0.80	0.93	0.85
L+SQ	0.94	0.82	0.81	0.93	0.85
L+S'Q	0.93	0.80	0.80	0.91	<b>0.90</b>
L+S'+ParQ	0.94	0.80	0.80	0.91	0.86
L+Q+S'Q	0.94	0.81	0.81	0.91	0.88
L+SQ+S'Q	<b>0.95</b>	<b>0.84</b>	<b>0.83</b>	<b>0.94</b>	0.88

Table 4: Dialog state updates classification accuracies where L stands for lexical features, Q - query overlap, P - prompt overlap, SQ - previous state query overlap, S'Q - new state query overlap, S'ParQ - new state parent query overlap.

## 6 The Complete Dialog System

Following the overall setup described in Section 2, we integrate the learned models into a complete dialog system. To evaluate the quality of the interactions with this system, we performed a small scale user study, as described here.

**Experimental Setup** We randomly chose 8 goals, across topics, from the 76 used to train our models. For each goal, six users were shown a Microsoft Word document before and after the goal was executed. Users were allowed to interact with the system in an unconstrained manner providing a total of 48 interactions. If at any point they thought the system response was irrelevant to their intent, they could either rephrase or start over.

**User Study** We objectively measured task completion and the quality of individual system utterances. We also asked the users to rate the system (on a scale of 1-5, with 5 being best) on various measures, including whether the system asked relevant questions. Table 5 summarizes the results. In general, users were able to complete the tasks and were reasonably satisfied.

**Discussion** Figure 6 shows four example dialogs. User utterances ranged from single keywords to rich, sentence-like goal descriptions. For the shorter inputs, the system was generally able to identify a small cohort of high-probability intents. For longer inputs, especially with explicitly specified parameter settings, the system was usually able to elicit whatever additional information was required to complete the task. The majority of errors fell into two categories when the system:

- I. missed key information in the input and asked redundant questions – Figure 6 (C);
- II. failed to correctly identify the intent, and asked an irrelevant question – Figure 6 (D).

Table 5 shows the frequency of each error type.

Objective Measures	Observed Rates	
User achieved task	83%	
System asked irrelevant questions	19%	
System asked redundant questions	18%	
User Satisfaction	Score	Std.Dev.
Overall satisfaction	3.42	0.49
Questions were relevant	4.17	0.41
Ambiguity was managed	4.00	0.89

Table 5: User study results.

U: “I want a page number and title at the bottom of the page”  
 S: “Choose a page number design from the gallery of designs”  
 U: \*User selects design\*  
 S: “To insert header or footer content before the page number, type the content, and then press TAB to position the content”  
 U: “Professional Resume”  
 S: “Task Completed”

(A)

U: “Alphabetize bullets”  
 S: “Do you want to sort a list alphabetically”  
 U: “Sort a list”  
 S: “Select the text in a bulleted or numbered list”  
 U: \*User selection\*  
 S: “Ascending or Descending?”  
 U: “Ascending”  
 S: “Task Completed”

(B)

U: “I want the background color to be yellow”  
 S: “Choose a color for the document”  
 U: “Yellow ”  
 S: “Task completed”

(C)

U: “I want to change the font face and size”  
 S: “Choose the default font”  
 U: “No, reduce the size of the text”  
 S: \*System Restarts\*

(D)

Figure 6: Four example dialogs from the user study, including cases that (A and B) complete successfully, (C) have a redundant question, and (D) fail to recognize the user intent.

## 7 Related work

To the best of our knowledge, this paper presents the first effort to induce full procedural dialog systems from instructional text and query click logs.

**Grounded Language Learning** There has been significant interest in grounded language learning. Perhaps the most closely related work learns to understand instructions and automati-

cally complete the tasks they describe (Branavan et al., 2009; Vogel and Jurafsky, 2010; Kushman et al., 2009; Branavan et al., 2010; Artzi and Zettlemoyer, 2013). However, these approaches did not model user interaction. There are also many related approaches for other grounded language problems, including understanding game strategy guides (Branavan et al., 2011), modeling users goals in a Windows domain (Horvitz et al., 1998), learning from conversational interaction (Artzi and Zettlemoyer, 2011), learning to sportscast (Chen and Mooney, 2011), learning from event streams (Liang et al., 2009), and learning paraphrases from crowdsourced captions of video snippets (Chen and Dolan, 2011).

**Dialog Generation from Text** Similarly to Piwek’s work (2007; 2010; 2011), we study extracting dialog knowledge from documents (monologues or instructions). However, Piwek’s approach generates static dialogs, for example to generate animations of virtual characters having a conversation. There is no model of dialog management or user interaction, and the approach does not use any machine learning. In contrast, to the best of our knowledge, we are the first to demonstrate it is possible to learn complete, interactive dialog systems using instructional texts (and non-expert annotation).

**Learning from Web Query Logs** Web query logs have been extensively studied. For example, they are widely used to represent user intents in spoken language dialogs (Tür et al., 2011; Celikyilmaz et al., 2011; Celikyilmaz and Hakkani-Tur, 2012). Web query logs are also used in many other NLP tasks, including entity linking (Pantel et al., 2012) and training product and job intent classifiers (Li et al., 2008).

**Dialog Modeling and User Simulation** Many existing dialog systems learn dialog strategies from user interactions (Young, 2010; Rieser and Lemon, 2008). Moreover, dialog data is often limited and, therefore, user simulation is commonly used (Scheffler and Young, 2002; Schatzmann et al., 2006; Georgila et al., 2005).

Our overall approach is also related to many other dialog management approaches, including those that construct dialog graphs from dialog data via clustering (Lee et al., 2009), learn information state updates using discriminative classification models (Hakkani-Tur et al., 2012; Mairesse et al.,

2009), optimize dialog strategy using reinforcement learning (RL) (Scheffler and Young, 2002; Rieser and Lemon, 2008), or combine RL with information state update rules (Heeman, 2007). However, our approach is unique in the use of inducing task and domain knowledge with light supervision to assist the user with many goals.

## 8 Conclusions and Future Work

This paper presented a novel approach for automatically constructing procedural dialog systems with light supervision, given only textual resources such as instructional text and search query click logs. Evaluations demonstrated highly accurate performance, on automatic benchmarks and through a user study.

Although we showed it is possible to build complete systems, more work will be required to scale the approach to new domains, scale the complexity of the dialog manager, and explore the range of possible textual knowledge sources that could be incorporated. We are particularly interested in scenarios that would enable end users to author new goals by writing procedural instructions in natural language.

## Acknowledgments

The authors would like to thank Jason Williams and the anonymous reviewers for their helpful comments and suggestions.

## References

- Priti Aggarwal, Ron Artstein, Jillian Gerten, Athanasios Katsamanis, Shrikanth Narayanan, Angela Nazarian, and David R. Traum. 2012. The twins corpus of museum visitor questions. In *Proceedings of LREC*.
- Yoav Artzi and Luke Zettlemoyer. 2011. Learning to recover meaning from unannotated conversational interactions. In *NIPS Workshop In Learning Semantics*.
- Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1(1):49–62.
- Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. 2010. Soyent: a word processor with a crowd inside. In *Proceedings of ACM Symposium on User Interface Software and Technology*.

- S. R. K. Branavan, Harr Chen, Luke S. Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proceedings of ACL*.
- S. R. K. Branavan, Luke S. Zettlemoyer, and Regina Barzilay. 2010. Reading between the lines: learning to map high-level instructions to commands. In *Proceedings of ACL*.
- S. R. K. Branavan, David Silver, and Regina Barzilay. 2011. Learning to win by reading manuals in a monte-carlo framework. In *Proceedings of ACL*.
- Asli Celikyilmaz and Dilek Hakkani-Tur. 2012. A joint model for discovery of aspects in utterances. In *Proceedings of ACL*.
- Asli Celikyilmaz, Dilek Hakkani-Tür, and Gokhan Tür. 2011. Mining search query logs for spoken language understanding. In *Proceedings of ICML*.
- David L. Chen and William B. Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of ACL*.
- David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of AAAI*.
- Myroslava Dzikovska, Amy Isard, Peter Bell, Johanna D. Moore, Natalie B. Steinhauser, Gwendolyn E. Campbell, Leanne S. Taylor, Simon Caine, and Charlie Scott. 2011. Adaptive intelligent tutorial dialogue in the beetle ii system. In *Proceedings of AIED*.
- Kallirroi Georgila, James Henderson, and Oliver Lemon. 2005. Learning user simulations for information state update dialogue systems. In *Proceedings of Eurospeech*.
- Dilek Hakkani-Tur, Gokhan Tur, Larry Heck, Ashley Fidler, and Asli Celikyilmaz. 2012. A discriminative classification-based approach to information state updates for a multi-domain dialog system. In *Proceedings of Interspeech*.
- Peter Heeman. 2007. Combining Reinforcement Learning with Information-State Update Rules. In *Proceedings of ACL*.
- Eric Horvitz, Jack Breese, David Heckerman, David Hovel, and Koos Rommelse. 1998. The Lumiere project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of Uncertainty in Artificial Intelligence*.
- Nate Kushman, Micah Brodsky, S. R. K. Branavan, Dina Katabi, Regina Barzilay, and Martin Rinard. 2009. WikiDo. In *ACM HotNets*.
- Cheongjae Lee, Sangkeun Jung, Kyungduk Kim, and Gary Geunbae Lee. 2009. Automatic agenda graph construction from human-human dialogs using clustering method. In *Proceedings of NAACL*.
- Xiao Li, Ye-Yi Wang, and Alex Acero. 2008. Learning query intent from regularized click graphs. In *Proceedings of SIGIR*.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of ACL-IJCNLP*.
- F. Mairesse, M. Gasic, F. Jurcicek, S. Keizer, B. Thomson, K. Yu, and S. Young. 2009. Spoken language understanding from unaligned data using discriminative classification models. In *Proceedings of Acoustics, Speech and Signal Processing*.
- Fabrizio Morbini, Eric Forbell, David DeVault, Kenji Sagae, David R. Traum, and Albert A. Rizzo. 2012. A mixed-initiative conversational dialogue system for healthcare. In *Proceedings of SIGDIAL*.
- Jorge Nocedal and Stephen J. Wright. 2000. *Numerical Optimization*. Springer.
- Patric Pantel, Thomas Lin, and Michael Gamon. 2012. Mining entity types from query logs via user intent. In *Proceedings of ACL*.
- Paul Piwek and Svetlana Stoyanchev. 2010. Generating expository dialogue from monologue: Motivation, corpus and preliminary rules. In *Proceedings of NAACL*.
- Paul Piwek and Svetlana Stoyanchev. 2011. Data-oriented monologue-to-dialogue generation. In *Proceedings of ACL*, pages 242–247.
- Paul Piwek, Hugo Hernault, Helmut Prendinger, and Mitsuru Ishizuka. 2007. T2d: Generating dialogues between virtual agents automatically from text. In *Proceedings of Intelligent Virtual Agents*.
- Verena Rieser and Oliver Lemon. 2008. Learning effective multimodal dialogue strategies from wizard-of-oz data: Bootstrapping and evaluation. In *Proceedings of ACL*.
- A. Rizzo, Kenji Sagae, E. Forbell, J. Kim, B. Lange, J. Buckwalter, J. Williams, T. Parsons, P. Kenny, David R. Traum, J. Difede, and B. Rothbaum. 2011. Simcoach: An intelligent virtual human system for providing healthcare information and support. In *Proceedings of ITSEC*.
- Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *Knowledge Engineering Review*, 21(2).
- Konrad Scheffler and Steve Young. 2002. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In *Proceedings of Human Language Technology Research*.
- Natalie B. Steinhauser, Gwendolyn E. Campbell, Leanne S. Taylor, Simon Caine, Charlie Scott, Myroslava Dzikovska, and Johanna D. Moore. 2011.

Talk like an electrician: Student dialogue mimicking behavior in an intelligent tutoring system. In *Proceedings of AIED*.

David R. Traum, Priti Aggarwal, Ron Artstein, Susan Foutz, Jillian Gerten, Athanasios Katsamanis, Anton Leuski, Dan Noren, and William R. Swartout. 2012. Ada and grace: Direct interaction with museum visitors. In *Proceedings of Intelligent Virtual Agents*.

Gökhan Tür, Dilek Z. Hakkani-Tür, Dustin Hillard, and Asli Çelikyılmaz. 2011. Towards unsupervised spoken language understanding: Exploiting query click logs for slot filling. In *Proceedings of Interspeech*.

Adam Vogel and Dan Jurafsky. 2010. Learning to follow navigational directions. In *Proceedings of ACL*.

Steve Young. 2010. Cognitive user interfaces. In *IEEE Signal Processing Magazine*.

# Public Dialogue: Analysis of Tolerance in Online Discussions

Arjun Mukherjee<sup>†</sup> Vivek Venkataraman<sup>†</sup> Bing Liu<sup>†</sup> Sharon Meraz<sup>‡</sup>

<sup>†</sup>Department of Computer Science <sup>‡</sup>Department of Communication  
University of Illinois at Chicago

arjun4787@gmail.com {vvenka6, liub, smeraz}@uic.edu

## Abstract

Social media platforms have enabled people to freely express their views and discuss issues of interest with others. While it is important to discover the topics in discussions, it is equally useful to mine the nature of such discussions or debates and the behavior of the participants. There are many questions that can be asked. One key question is whether the participants give reasoned arguments with justifiable claims via constructive debates or exhibit dogmatism and egotistic clashes of ideologies. The central idea of this question is *tolerance*, which is a key concept in the field of communications. In this work, we perform a computational study of tolerance in the context of online discussions. We aim to identify tolerant vs. intolerant participants and investigate how disagreement affects tolerance in discussions in a quantitative framework. To the best of our knowledge, this is the first such study. Our experiments using real-life discussions demonstrate the effectiveness of the proposed technique and also provide some key insights into the psycholinguistic phenomenon of tolerance in online discussions.

## 1 Introduction

Social media platforms have enabled people from anywhere in the world to express their views and discuss any issue of interest in online discussions/debates. Existing works in this context include recognition of support and oppose camps (Agrawal et al., 2003), mining of authorities and subgroups (Mayfield and Rosè, 2011; Abu-Jbara et al. (2012), dialogue act segmentation and classification (Morbini and Sagae, 2011; Boyer et al., 2011), etc.

This paper probes further to study a different and important angle, i.e., the psycholinguistic phenomenon of *tolerance* in online discussions. Tolerance is an important concept in the field of communications. It is a subfacet of deliberation which refers to critical thinking and exchange of rational arguments on an issue among participants that seek to achieve consensus/solution

(Habermas, 1984).

Perhaps the most widely accepted definition of tolerance is that of Gastil (2005; 2007), who defines tolerance as a means to engage (in written or spoken communication) in critical thinking, judicious argument, sound reasoning, and justifiable claims through constructive discussion as opposed to mere coercion/egotistic clashes of ideologies.

In this work, we adopt this definition, and also employ the following characteristics of tolerance (also known as “code of conduct”) (Crocker, 2005; Gutmann and Thompson, 1996) to guide our work.

*Reciprocity*: Each member (or participant) offers proposals and justifications in terms that others could understand and accept.

*Publicity*: Each member engages in a process that is transparent to all and each member knows with whom he is agreeing or disagreeing.

*Accountability*: Each member gives acceptable and sound reasons to others on the various claims or proposals suggested by him.

*Mutual respect and civic integrity*: Each member’s speech should be morally acceptable, i.e., using proper language irrespective of agreement or disagreement of views.

The issue of tolerance has been actively researched in the field of communications for the past two decades, and has been investigated in multiple dimensions. However, existing studies are typically qualitative and focus on theorizing the socio-linguistic aspects of tolerance (more details in §2).

With the rapid growth of social media, the large volumes of online discussions/debates offer a golden opportunity to investigate people’s implicit psyche in discussions quantitatively based on the real-life data, i.e., their tolerance levels and their arguing nature, which are of fundamental interest to several fields, e.g., communications, marketing, politics, and sociology (Dahlgren, 2005; Gastil, 2005; Moxey and

Sanford, 2000). Communication and political scholars are hopeful that technologies capable of identifying tolerance levels of people on social issues (often discussed in online discussions) can render vital statistics which can be used in predicting political outcomes in elections and helpful in tailoring voting campaigns and agendas to maximize winning chances (Dahlgren, 2002).

**Objective:** The objective of this work is two-fold:

1. Identifying tolerant and intolerant participants in discussions.
2. Analyzing how disagreement affects tolerance and estimating the tipping point of such effects.

To the best of our knowledge, these tasks have not been attempted quantitatively before. The first task is a classification/prediction problem. Due to the complex and interactive nature of discussions, the traditional n-gram features are no longer sufficient for accurate classification. We thus propose a generative model, called DTM, to discover some key pieces of information which characterize the nature of discussions and their participants, e.g., the arguing nature (agreeing vs. disagreeing), topic and expression distributions. These allow us to generate a set of novel features from the estimated latent variables of DTM capable of capturing authors' tolerance psyche during discussions. The features are then used in learning to identify tolerant and intolerant authors. Our experimental results show that the proposed approach is effective and outperforms several strong baselines significantly.

The second task studies the interplay of tolerance and disagreement. It is well-known that tolerance facilitates constructive disagreements, but sustained disagreements often result in a transition to destructive disagreement leading to polarization and intolerance (Dahlgren, 2005). An interesting question is: What is the tipping point of disagreement to exhibit intolerance? We take a Bayesian approach to seek an answer and discover issue-specific tipping points. Our empirical results discover some interesting relationships which are supported by theoretical studies in psychology and linguistic communications.

Finally, this work also produces an annotated corpus of tolerant and intolerant users in online discussions across two domains: politics and religion. We believe this is the first such dataset and will be a valuable resource to the community.

## 2 Related Work

Although limited work has been done on analysis of tolerance in online discussions, there are several general research areas that are related to our work.

**Communications:** Tolerance has been an active research area in the field of communications for the past two decades. Ryfe (2005) provided a comprehensive survey of the literature. The topic has been studied in multiple dimensions, e.g., opinion and attitude (Luskin et al., 2004; Price et al., 2002), public engagement (Escobar, 2012), psychoanalysis (Slavin and Kriegman, 1992), argument repertoire (Cappella et al., 2002), etc.

Tolerance has also been investigated in the domain of political communications with an emphasis on political sophistication (Gastil and Dillard, 1999), civic culture (Dahlgren, 2002), and democracy (Fishkin, 1991). These existing works study tolerance from the qualitative perspective. Our focus is quantitative analysis.

**Sentiment analysis:** Sentiment analysis determines positive or negative opinions expressed on topics (Liu, 2012; Pang and Lee, 2008). Main tasks include aspect extraction (Hu and Liu, 2004; Popescu and Etzioni, 2005; Mukherjee and Liu, 2012c; Chen et al., 2013), opinion polarity identification (Hassan and Radev, 2010; Choi and Cardie, 2010) and subjectivity analysis (Wiebe, 2000). Although related, tolerance is different from sentiment. Sentiments are mainly indicated by sentiment terms (e.g., *great*, *good*, *bad*, and *poor*). Tolerance in discussions refers to the reception of certain views and often indicated by agreement and disagreement expressions and other features (§5).

**Online discussions or debates:** Several works put authors in debate into support and oppose camps. Agrawal et al. (2003) used a graph based method, and Murakami and Raymond (2010) used a rule-based method. In (Mukherjee and Liu, 2012a), contention points were identified, in (Mukherjee and Liu, 2012b), various expressions in review comment discussions were mined, and in (Galley et al., 2004; Hillard et al., 2003), speaker utterances were classified into agreement, disagreement, and backchannel classes. Also related are studies on linguistic style accommodation (Mukherjee and Liu, 2012d) and user pair interactions (Mukherjee and Liu, 2013) in online debates. However, these works do not consider tolerance analysis in debate discussions, which is the focus of this work.

In a similar vein, several classification methods have been proposed to recognize opinion stances and speaker sides in online debates (Somasundaran and Wiebe, 2009; Thomas et al., 2006; Bansal et al., 2008; Burfoot et al., 2011; Yessenalina et al., 2010). Lin and Hauptmann (2006) also proposed a method to identify opposing perspectives. Abu-Jbara et al. (2012) identified subgroups. Kim and Hovy (2007) studied election prediction by analyzing online discussions. Other related works studying dialogue and discourse in discussions include authority recognition (Mayfield and Rosè, 2011), dialogue act segmentation and classification (Morbini and Sagae, 2011; Boyer et al., 2011), discourse structure prediction (Wang et al., 2011).

All these prior works are valuable. But they are not designed to identify tolerance or to analyze tipping points of disagreements for intolerance in discussions which are the focus of this work.

### 3 Discussion/Debate Data

For this research, we used discussion posts from Volconvo.com. This forum is divided into various domains: Politics, Religion, Science, etc. Each domain consists of multiple discussion threads. Each thread consists of a list of posts. Our experimental data is from two domains, Politics and Religion. The data is summarized in Table 1(a). In this work, the terms users, authors and participants are used interchangeably. The full data is used for modeling, but 436 and 501 authors from Politics and Religion domains were manually labeled as being tolerant or intolerant (Table 1(c)) respectively for classification experiments.

Two judges (graduate students) were used to label the data. The judges are fluent in English and were briefed on the definition of tolerance (see §1). From each domain (Politics, Religion), we randomly sampled authors having not more than 60 posts in order to reduce the labeling burden as the judges need to read all posts and see all interactions of each author before providing a label. Given all posts by an author,  $a$  and his/her associated interactions (posts by other authors replying or quoting  $a$ ), the judges were asked to provide a label for author  $a$  as being *tolerant* or *intolerant*. In our labeling, we found that users strongly exhibit one dominant trait: tolerant or intolerant, as our data consists of topics like elections, immigration, theism, terrorism, and vegetarianism across politics and religion domains,

Domain	Posts	Authors	Cohen's $\kappa$	Tol.	Intol.	Total
Politics	48605	1027	0.74	213	223	436
Religion	66835	1370	0.77	207	294	501

(a) Full Data (b) Agreement (c) Labeled data

Table 1: Data statistics (Tol: Tolerant users; Intol: Intolerant users. Total = Tol. + Intol).

which are often heated and thus attract people with pre-determined, strong, and polarized stances<sup>1</sup>.

The judges worked in isolation (to prevent bias) during annotation/labeling and were also asked to provide a short reason for their judgment. The agreement statistics using Cohen's kappa are given in Table 1(b), which shows substantial agreements according to the scale<sup>2</sup> in (Landis and Koch, 1977). This shows that tolerance as defined in §1 is quite decisive and one can decide whether a debater is exhibiting tolerant vs. intolerant quite well. To account for disagreements in labels, the judges discussed their reasons to reach a consensus. The final labeled data is reported in Table 1(c).

### 4 Model

We now present our generative model to capture the key aspects of discussions/debates and their intricate relationships, which enable us to (1) design sophisticated features for classification and (2) perform an in-depth analysis of the interplay of disagreement and tolerance. The model is called Debate Topic Model (DTM).

DTM is a semi-supervised generative model motivated by the joint occurrence of various topics; and agreement and disagreement expressions (abbreviated AD-expressions hereon) in debate posts. A typical debate post mentions a few topics (using similar topical terms) and expresses some viewpoints with one or more AD-expression types (Agreement and Disagreement) using semantically related expressions. This observation forms the basis of the generative process of our model where documents (posts) are represented as admixtures of latent topics and AD-expression types (Agreement and Disagreement). This key observation and the motivation of modeling debates are from our previous work in (Mukherjee and Liu, 2012a). In the new set-

<sup>1</sup> These hardened perspectives are theoretically supported by the polarization effect (Sunstein, 2002), and the hostile media effect, a scenario where partisans rigidly hold on to their stances (Hansen and Hyunjung, 2011).

<sup>2</sup> Agreement levels are as follows.  $\kappa \in [0, 0.2]$ : Poor,  $\kappa \in (0.2, 0.4]$ : Fair,  $\kappa \in (0.4, 0.6]$ : Moderate,  $\kappa \in (0.6, 0.8]$ : Substantial, and  $\kappa \in (0.8, 1.0]$ : Almost perfect agreement.

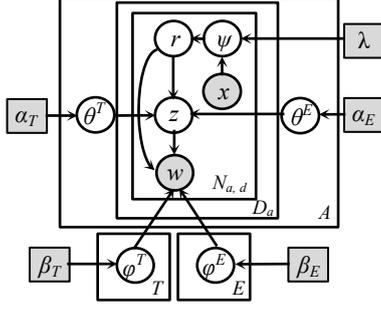


Figure 1: Plate notation of DTM

Variable/Function	Description
$a; A; d$	An author $a$ ; set of all authors; document, $d$
$(a, d); D_a$	Post $d$ by author $a$ ; Set of all posts by $a$
$T; E; V$	# of topics; expression types; vocabulary
$w_{a,d,j}; N_{a,d}$	$j^{\text{th}}$ term in $(a, d)$ ; Total # of terms in $(a, d)$
$\psi_{a,d,j}$	Distribution over topics and AD-expressions
$x_{a,d,j}$	Associated feature context of observed $w_{a,d,j}$
$\lambda$	Learned Max-Ent parameters
$r_{a,d,j} \in \{\hat{t}, \hat{e}\}$	Binary indicator/switch variable (topic ( $\hat{t}$ ) or AD-expression ( $\hat{e}$ )) for $w_{a,d,j}$
$\theta_a^T; \theta_a^E(\theta_{a,Ag}^E, \theta_{a,DisAg}^E)$	$a$ 's distribution over topics; expression types (Agreement: $\theta_{a,Ag}^E$ , Disagreement: $\theta_{a,DisAg}^E$ )
$\theta_{a,d}^T; \theta_{a,d}^E$	Topic distribution of post $d$ by author $a$ ; Probability mass of topic $t$ in $\theta_{a,d}^T$ .
$\theta_{a,d,e}^E; \theta_{a,d,e \in \{Ag, DisAg\}}^E$	Expression type distribution of post $d$ by author $a$ ; Corresponding probability masses of Agreement: $\theta_{a,d,e=Ag}^E$ and Disagreement in $\theta_{a,d,e=DisAg}^E$ .
$z_{a,d,j}$	Topic/Expression type of $w_{a,d,j}$
$\phi_t^T; \phi_e^E$	Topic $t$ 's; Expression type $e$ 's distribution over vocabulary terms
$\alpha^T; \alpha^E; \beta^T; \beta^E$	Dirichlet priors of $\theta_a^T; \theta_a^E; \phi_t^T; \phi_e^E$
$n_{a,t}^{AT}; n_{a,e}^{AE}$	# of times topic $t$ ; expression type $e$ assigned to $a$
$n_{t,v}^{TV}; n_{e,v}^{EV}$	# of times term $v$ appears in topic $t$ ; expression type $e$

Table 2: List of notations

ting, we model topics and debate expression distributions specific to authors as this work is concerned with modeling authors' (in)tolerance nature. Making latent variable  $\theta^E$  and  $\theta^T$  author specific facilitates modeling user behaviors (§5.3).

Assume we have  $t_{1..T}$  topics and  $e_{1..E}$  expression types in our corpus. In our case of debate posts, based upon reading various posts, we hypothesize that  $E = 2$  as in debates as we mostly find 2 dominant expression types: Agreement and Disagreement. Meanings of variables used in the following discussion are detailed in Table 2.

In this work, a document/post is viewed as a bag of  $n$ -grams and we use terms to denote both words (unigrams) and phrases ( $n$ -grams)<sup>3</sup>. DTM is a switching graphical model performing a switch between topics and AD-expressions similar to that in (Zhao et al., 2010). The switch is done using a learned maximum entropy (Max-Ent) model. The rationale here is that topical and AD-expression terms usually play different syntactic roles in a sentence. Topical terms (e.g., "U.S. elections," "government," "income tax") tend to be noun and noun phrases while expression terms ("I refute," "how can you say," "I'd agree") usually contain pronouns, verbs, wh-determiners, and modals. In order to utilize the part-of-speech (POS) tag information, we place the topic/AD-expression distribution,  $\psi_{a,d,j}$  (the prior over the indicator variable  $r_{a,d,j}$ ) in the term plate (Figure 1) and set it using a Max-Ent model conditioned on the observed context  $x_{a,d,j}$  associated with  $w_{a,d,j}$  and the learned Max-Ent parameters  $\lambda$  (details in §4.1). In this work, we use both lexical and POS features of the previous, current and next POS tags/lexemes of the term  $w_{a,d,j}$  as the contextual information, i.e.,

$$x_{a,d,j} = [POS_{w_{a,d,j-1}}, POS_{w_{a,d,j}}, POS_{w_{a,d,j+1}}, w_{a,d,j-1}, w_{a,d,j}, w_{a,d,j+1}],$$

which is used to produce feature functions for Max-Ent. For phrasal terms ( $n$ -grams), all POS tags and lexemes of  $w_{d,j}$  are considered as contextual information for computing feature functions in Max-Ent. DTM has the following generative process:

- A. For each AD-expression type  $e$ , draw  $\phi_e^E \sim \text{Dir}(\beta^E)$
- B. For each topic  $t$ , draw  $\phi_t^T \sim \text{Dir}(\beta^T)$
- C. For each author  $a \in \{1 \dots A\}$ :
  - i. Draw  $\theta_a^E \sim \text{Dir}(\alpha^E)$
  - ii. Draw  $\theta_a^T \sim \text{Dir}(\alpha^T)$
  - iii. For each document/post  $d \in \{1 \dots D_a\}$ :
    - I. For each term  $w_{a,d,j}, j \in \{1 \dots N_{a,d}\}$ :
      - a. Set  $\psi_{a,d,j} \leftarrow \text{MaxEnt}(x_{a,d,j}; \lambda)$
      - b. Draw  $r_{a,d,j} \sim \text{Bernoulli}(\psi_{a,d,j})$
      - c. if ( $r_{a,d,j} = \hat{e}$ ) //  $w_{a,d,j}$  is an AD-expression term  
Draw  $z_{a,d,j} \sim \text{Mult}(\theta_{a,d,j}^E)$   
else //  $r_{a,d,j} = \hat{t}$ ,  $w_{a,d,j}$  is a topical term  
Draw  $z_{a,d,j} \sim \text{Mult}(\theta_a^T)$
      - d. Emit  $w_{a,d,j} \sim \text{Mult}(\phi_{z_{a,d,j}}^{r_{a,d,j}})$

#### 4.1 Inference

We employ posterior inference using Monte Car-

<sup>3</sup> Topics in most topic models (e.g., LDA (Blei et al., 2003)) are unigram distributions and a document is treated as an exchangeable bag-of-words. This offers a computational advantage over models considering word orders (Wallach, 2006). As our goal is to enhance the expressiveness of DTM (rather than "modeling" word order), we use 1-4 grams preserving the advantages of exchangeable modeling.

Disagreement expressions ( $\varphi_{e=Disagreement}^E$ )
I, disagree, I don't, I disagree, <b>argument</b> , reject, <b>claim</b> , I reject, I refute, <b>and</b> , <b>your</b> , I refuse, <b>won't</b> , <b>the claim</b> , nonsense, <i>I contest</i> , dispute, <b>I think</b> , completely disagree, don't accept, don't agree, incorrect, <b>doesn't</b> , <i>hogwash</i> , <i>I don't buy your</i> , <i>I really doubt</i> , your nonsense, <b>true</b> , <i>can you prove</i> , argument fails, <i>you fail to</i> , <b>your assertions</b> , <i>bullshit</i> , <i>sheer nonsense</i> , <i>doesn't make sense</i> , <i>you have no clue</i> , <i>how can you say</i> , <i>do you even</i> , <i>contradict yourself</i> , ...
Agreement expressions ( $\varphi_{e=Agreement}^E$ )
agree, <b>I</b> , correct, yes, true, accept, I agree, <b>don't</b> , indeed correct, <b>your</b> , <b>point</b> , <b>that</b> , I concede, is valid, <b>your claim</b> , <b>not really</b> , <i>would agree</i> , <b>might</b> , <i>agree completely</i> , yes indeed, absolutely, you're correct, <i>valid point</i> , <b>argument</b> , <b>the argument</b> , proves, <i>do accept</i> , support, agree with you, <i>rightly said</i> , <b>personally</b> , well put, <i>I do support</i> , <i>personally agree</i> , <b>doesn't necessarily</b> , exactly, <i>very well put</i> , absolutely correct, <i>kudos</i> , <i>point taken</i> , ...

Table 3: Top terms (comma delimited) of two expression types. **Red (bold)** terms denote possible errors. *Blue (italics)* terms are newly discovered; rest (black) terms have been used in Max-Ent training.

lo Gibbs sampling. Denoting the random variables  $\{w, z, r\}$  by singular scripts  $\{w_k, z_k, r_k\}, k_{1...K}$ , where  $K = \sum_a \sum_d N_{a,d}$ , a single iteration consists of performing the following sampling:

$$p(z_k = t, r_k = \hat{t} | W_{-k}, Z_{-k}, R_{-k}, w_k = v) \propto \frac{\exp(\sum_{i=1}^n \lambda_i f_i(x_{a,d,j}, \hat{t}))}{\sum_{y \in \{\hat{t}, \bar{t}\}} \exp(\sum_{i=1}^n \lambda_i f_i(x_{a,d,j}, y))} \times \frac{n_{a,t,-k}^{AT} + \alpha^T}{n_{a,(\cdot)-k}^{AT} + T\alpha^T} \times \frac{n_{t,v,-k}^{TV} + \beta^T}{n_{t,(\cdot)-k}^{TV} + V\beta^T} \quad (1)$$

$$p(z_k = e, r_k = \hat{e} | W_{-k}, Z_{-k}, R_{-k}, w_k = v) \propto \frac{\exp(\sum_{i=1}^n \lambda_i f_i(x_{a,d,j}, \hat{e}))}{\sum_{y \in \{\hat{e}, \bar{e}\}} \exp(\sum_{i=1}^n \lambda_i f_i(x_{a,d,j}, y))} \times \frac{n_{a,e,-k}^{AE} + \alpha^E}{n_{a,(\cdot)-k}^{AE} + E\alpha^E} \times \frac{n_{e,v,-k}^{EV} + \beta^E}{n_{e,(\cdot)-k}^{EV} + V\beta^E} \quad (2)$$

where  $k = (a, d, j)$  denotes the  $j^{th}$  term of document  $d$  by author  $a$  and the subscript  $-k$  denotes assignments excluding the term at  $(a, d, j)$ . Omission of the latter index denoted by  $(\cdot)$  represents the marginalized sum over the latter index. Count variables are detailed in Table 1 (last two rows).  $\lambda_{1...n}$  are the parameters of the learned Max-Ent model corresponding to the  $n$  binary feature functions  $f_{1...n}$  for Max-Ent. The learned Max-Ent  $\lambda$  parameters in conjunction with the observed context,  $x_{a,d,j}$  feed the supervision signal for updating the topic/expression switch parameter,  $r$  in equations (1) and (2).

The hyper-parameters for the model were set to the values  $\beta^T = \beta^E = 0.1$  and  $\alpha_T = 50/T$ ,  $\alpha_E = 50/E$ , suggested in (Griffiths and Steyvers, 2004). Model parameters were estimated after 5000 Gibbs iterations with a burn-in of 1000 iterations. The Max-Ent parameters  $\lambda$  were learned using 500 labeled terms in each domain (politics:- topical: 376 and AD-expression: 124; religion:- topical: 349 and AD-expression: 151) appearing at least 10 times in debate threads other than the data in Table 1 (we do so since the

data in Table 1(c) is later used in the classification experiments in §6.1).

Table 3 lists some top AD-expressions discovered by DTM. We see that DTM can cluster many correct AD-expressions, e.g., “I disagree”, “I refute”, “don’t accept”, etc. in disagreement; and “I agree”, “you’re correct”, “agree with you”, etc. in agreement. Further, it also discovers highly specific and more distinctive expressions beyond those used in Max-Ent training (marked *blue* in italics), e.g., “I don’t buy your”, “can you prove”, “you fail to”, and “you have no clue” in disagreement; and phrases like “valid point”, “rightly said”, “I do support”, and “very well put” in agreement. In §6.1, we will see that these AD-expressions serve as high quality features for predicting tolerance.

Lastly, we note that DTM also estimates several pieces of useful information (e.g., AD-expressions, posterior estimates of author’s arguing nature,  $\theta_a^E$ ; latent topics and expressions,  $\varphi_t^T$ ;  $\varphi_e^E$ , etc.). These will be used to produce a rich set of user behavioral features for characterizing tolerance in §5.3.

## 5 Feature Engineering

We now propose features which will be used for model building to classify tolerant and intolerant authors in Table 1(c). We use three sets of features.

### 5.1 Language based Features of Tolerance

**Word and POS n-grams:** As tolerance in communication is directly reflected in language usage, word  $n$ -grams are obvious features. We also use POS tags (obtained using Stanford Tagger<sup>4</sup>) as features. The rationale of using POS tag based features is that intolerant communications are often characterized by hate/egotistic speech which have pronounced use of specific part of speech (e.g., pronouns) (Zingo, 1998).

**Heuristic Factor Analysis:** In psycholinguistics, factor analysis refers to the process of finding groups of semantically similar linguistic constructs (words/phrases). It is also called meaning extraction in (Chung and Pennebaker, 2007). As tolerance in discussions is characterized by reasoned expressions which often accompany sourcing (e.g., providing a hyperlink, making an attempt to clarify with some evidence, etc.), we compiled a list of reasoned and sourced expressions (shown in Table 4) from prior works

<sup>4</sup> <http://nlp.stanford.edu/software/tagger.shtml>

Factor: Reasoning words/phrases
because, because of, since, reason, reason being, reason is, reason why, due to, owing to, as in, therefore, thus, henceforth, hence, implies, implies that, implying, hints, hinting, hints towards, it follows that, it turns out, conclude, consequence, consequently, the cause, rationale, the rationale, justification, the justification, provided, premise, assumption, on the proviso, in spite, ...
Factor: Sourcing words/phrases
<i>presence of hyperlinks/urls</i> , source, reference, for example, for instance, namely, to explain, to detail, to clarify, to elucidate, to illustrate, to be precise, furthermore, moreover, apart from, besides, we find, ...

Table 4: Heuristic Factor Analysis (HFA). Words/Phrases in each factor compiled from prior works in psycholinguistics.

(Chung and Pennebaker, 2007; Flor and Hadar, 2005; Moxey and Sanford, 2000; Pennebaker, et al., 2007).

## 5.2 Debate Expression Features

**AD-expressions:** As we have seen in §4, DTM can discover specific agreement and disagreement expressions in debates. We use these expressions as another feature set. Estimated AD-expressions (Table 3) serve as a principled way of performing factor analysis in debates instead of heuristic factor analysis as in Table 4 used in prior works.

As the AD-expression types are modeled as Dirichlet distributions ( $\varphi^E \sim \text{Dir}(\beta^E)$ ), due to the smoothing effect, each term in the vocabulary has some non-zero probability mass associated with the expression types. To ensure that the discovered expressions are representative AD-expressions, we only consider the terms in  $\varphi^E$  with  $p(v|e) = \varphi_{e,v}^E > 0.001$  as probability masses lower than 0.001 are more due to the smoothing effect of Dirichlet distribution than true correlation.

## 5.3 User Behavioral Features

Here we propose several features of user interaction which reflect the socio-psychological state of tolerance while participating in discussions. We note that these features rely on the posterior estimates of latent variables  $\theta^E$ ,  $z$ , and  $r$  in DTM (§4) and are thus difficult to obtain without modeling.

**Overall Arguing Nature:** The posterior on  $\theta_a^E$  (Table 2) for each author,  $a$  gives an estimate of  $a$ 's overall arguing nature (agreeing or disagreeing). We use the probability mass assigned to each arguing nature type as a user behavioral feature. This gives us two features  $f_1, f_2$  as fol-

lows:

$$f_1(a) = \theta_{a,Ag}^E; f_2(a) = \theta_{a,DisAg}^E \quad (3)$$

**Behavioral Response:** As intolerant users are likely to attract more disagreement, it is naturally useful to estimate the response (agreeing vs. disagreeing) a user receives from other users. For computing behavioral response, we first use the posterior on  $z$  to compute the distribution of AD-expressions (i.e., the relative probability masses of agreeing and disagreeing expressions) in a document  $d$  by an author  $a$  as follows:

$$\theta_{a,d,Ag}^E = \frac{|\{j|z_{a,d,j}=Ag, 1 \leq j \leq N_{a,d}\}|}{|\{j|r_{a,d,j}=\hat{e}, 1 \leq j \leq N_{a,d}\}|},$$

$$\theta_{a,d,DisAg}^E = \frac{|\{j|z_{a,d,j}=DisAg, 1 \leq j \leq N_{a,d}\}|}{|\{j|r_{a,d,j}=\hat{e}, 1 \leq j \leq N_{a,d}\}|} \quad (4)$$

Now to get the overall behavioral response of an author,  $a$  we take the expected value of the agreeing and disagreeing responses that  $a$  received from other authors  $a'$  who replied to or quoted  $a$ 's posts. The expectations below are taken over all posts  $d'$  by  $a'$  which reply/quote posts of  $a$ .

$$f_3(a) = E[\theta_{a',d',Ag}^E]; f_4(a) = E[\theta_{a',d',DisAg}^E] \quad (5)$$

**Equality of Speech:** In communication literature (Dahlgren, 2005; Habermas, 1984), equality is theorized as an essential element of tolerance. Each participant must be able to participate on an equal footing with others without anybody dominating the discussion. In online debates, we can measure this phenomenon using the following feature:

$$f_5(a) = E\left[\left(\frac{\# \text{ of posts by } a \text{ in thread } l}{\# \text{ of posts in thread } l}\right) E[\theta_{a,d,DisAg}^E]\right] \quad (6)$$

where the inner expectation is taken over all posts of  $a$  in thread  $l$  and the outer expectation is taken over all threads  $l$  in which  $a$  participated. The above definition computes the aggressive posting behavior of author  $a$  whereby he tries to dominate the thread by posting more than others. The aggressive posting behavior is weighted by author's disagreeing nature because a person usually exhibits a dominating nature when he pushes hard to establish his ideology (which is often in disagreement with others) (Moxey and Sanford, 2000).

**Topic Shifts:** An interesting phenomenon of human (social) psyche is that when people are unable to logically argue their stances and feel they are losing the debate, they often try to belittle/deride others by pulling unrelated topics into discussion (Slavin and Kriegman, 1992). This is

referred to as topic shifts. Topic shifts thus have a relation with tolerance in deliberation. Stromer-Galley (2005) reported that if the discussion is off topic, then tolerance or deliberation cannot meet its objective of deep consideration of an issue. Hence, the average topic shifts of an author,  $a$  across various posts in a thread can serve as a good feature for measuring tolerance. We use the posterior on per-document topic distribution,  $\theta_{a,d,t}^T = \frac{|\{j|z_{a,d,j}=t, 1 \leq j \leq N_{a,d}\}|}{|\{j|r_{a,d,j}=\hat{t}, 1 \leq j \leq N_{a,d}\}|}$  to measure topic shifts using KL-Divergence as follows:

$$f_6 = E \left[ \text{avg}_{d,d' \in \text{thread } l} \left( D_{KL}(\theta_{a,d}^T || \theta_{a,d'}^T) \right) \right] \quad (7)$$

We first compute author,  $a$ 's average topic shifts in a thread,  $l$  which measures his topic shifts in  $l$ . But this only gives us his behavior in one thread. To capture his overall behavior, we take the expected value of this behavior over all threads in which  $a$  participated. We take average KL-divergence (KL-Div.) over all pairs of posts by  $a$  in a given thread to account for the asymmetry of KL-Div.

Finally, we note that by no means do we claim that the mere presence and a large value of any of the above features imply that a user is intolerant or tolerant. They are indicators of the phenomenon of tolerance in discussions/debates. The actual prediction is done using the learned models in §6.1.

## 6 Experimental Evaluation

We now detail the experiments that investigate the strengths of features in §5. In particular, we first consider the task of classifying whether an author is tolerant or intolerant in discussions. Then, we analyze how disagreement affects tolerance.

### 6.1 Tolerant and Intolerant Classification

Here, we show that the features in §5 can help build accurate models for predicting tolerance. We employ a linear kernel<sup>5</sup> SVM (using the SVM<sup>Light</sup> system (Joachims, 1999)) and report 5-fold cross validation (CV) results on the task of predicting the socio-psychological nature of users' communication: tolerant vs. intolerant in politics and religion domains (Table 1(c)). Note that for each fold of 5-fold CV, DTM was run on the full data of each domain (Table 1(a)) excluding the users (and their associated posts) in the test set of that fold for generating the features of the training instances (users). The learned DTM

<sup>5</sup> Other kernels (rbf, poly, sigmoid) did not perform as well.

was then fitted (using the approach in (Hofmann, 1999)) to the test set users and their posts for generating the features of the test instances.

To investigate the effectiveness of the proposed framework, we incrementally add feature sets starting with the baseline features. Word unigrams and bigrams (inclusive of unigrams)<sup>6</sup> serve as our first baseline (B1a, B1b). Word + POS bigrams is our second baseline (B2). "Word" in B2 uses bigrams as B1b gives better results. B2 + Heuristic Factor Analysis (HFA) (Table 4) serve as our third baseline (B3). Table 5 shows the experiment results. We note the following:

1. Across both domains, adding POS bigrams slightly improves classification accuracy and F<sub>1</sub>-score beyond standard word unigrams and bigrams. Feature selection using information gain (IG) does not help much.
2. Using heuristic factor analyses (HFA) of reasoned and sourced expressions (Table 4) brings about 1% and 2% improvement in accuracy in politics and religion domains respectively.
3. Debate expression features (DE) in §5.2 and user behavioral features (UB) in §5.3 produced from DTM progressively improve classification accuracies by 4% and 8% in politics domains and 5% and 6% in religion domains. The improvements are also statistically significant.

In summary, we can see that modeling made a major impact. It improved the accuracy by about 10% than traditional unigram and bigram baselines. This shows that the debate expressions and user behaviors computed using the DTM model can capture various dimensions of (in)tolerance not captured by n-grams.

### 6.2 How Disagreement affects Tolerance?

We now quantitatively study the effect of disagreement on tolerance. We recall from §1 that tolerance indicates constructive discussion and allows disagreement. Some level of disagreement is often times an integral component of deliberation and tolerance (Cappella et al., 2002).

Disagreements, however, can be either constructive or destructive. The distinction is that the former is aimed at arriving at a consensus or solution, while the latter leads to polarization and intolerance (Sunstein, 2002). It was also shown in (Dahlgren, 2005) that sustained disa-

<sup>6</sup> Higher order  $n$ -grams did not result in better results.

Feature Setting	Politics				Religion			
	Precision	Recall	F <sub>1</sub>	Accuracy	Precision	Recall	F <sub>1</sub>	Accuracy
B1a: Word unigrams	64.1	86.3	73.7	70.1	61.9	86.8	72.6	71.9
Word unigram + IG	64.5	86.2	73.9	70.2	62.7	86.9	72.9	71.9
B1b: Word bigrams	66.8	87.8	75.9	72.4	64.9	89.1	75.9	75.1
B2: W+POS bigrams	68.5	86.8	76.4	73.7	66.6	88.4	76.8	76.7
B3: B2 + HFA(Table 4)	69.2	90.5	78.1	75.2	66.4	90.6	76.8	77.5
B3 + DE (§5.2)	74.7	91.3	<b>82.4†</b>	<b>79.5†</b>	70.2	92.8	<b>80.8†</b>	<b>82.1†</b>
B3 + DE + UB (§5.3)	76.1	92.2	<b>83.1‡</b>	<b>83.2‡</b>	71.7	93.4	<b>82.1‡</b>	<b>83.3‡</b>

Table 5: Precision, Recall, F<sub>1</sub> score on the tolerant class, and Accuracy for different feature settings across 2 domains. DE: Debate expression features (AD-expressions, Table3, §5.2). UB: User behavioral features (§5.3). Improvements in F<sub>1</sub> and Accuracy using DTM features (beyond baselines, B1-B3) are statistically significant (†:  $p < 0.02$ ; ‡:  $p < 0.01$ ) using paired  $t$ -test with 5-fold CV.

Thread/Issue	# Posts	# Users	% InTol.	$E[\theta_{a,d,DisAg}^E]$	$\tau$	$p$ -value
Repeal Healthcare	1823	33	39.9	0.57	0.65	0.02
Europe’s Collapse	1824	33	42.5	0.61	0.61	0.01
Obama Euphoria	1244	26	30.7	0.66	0.71	0.01
Socialism	831	49	44.8	0.69	0.48	0.03
Abortion	1232	58	48.4	0.78	0.37	0.01

Table 6: Tipping points of disagreements for intolerance ( $\tau$ ) of different issues.  $E[\theta_{a,d,DisAg}^E]$ : the expected disagreement over all posts in each issue/thread, # Posts: the total number of posts, # Users: the total number of users/authors, % Intol: % of intolerant users in each thread,  $\tau$ : the estimated tipping point, and  $p$ -value: computed from two-tailed Fisher’s exact test.

greement often takes a transition towards destructive disagreement and is likely to lead to intolerance. Similar phenomena was also identified in psychology literature (Critchley, 1964). In such cases, the participants often stubbornly stick to an extreme attitude, which eventually results in intolerance and defeats the very purpose of deliberative discussion.

An intriguing research question is: What is the relationship between disagreement and intolerance? The question is interesting from both the communication and psycholinguistic perspectives. The best of our knowledge, this is the first attempt towards seeking an answer. We work in the context of five issues/threads in real-life online debates. To derive quantitative and definite conclusions, it is required to perform the following tasks:

- For each issue, empirically investigate *in expectation* the *tipping point* of disagreement beyond which a user tends to be intolerant.
- Further, investigate the *confidence* on the estimated tipping point (i.e., what is the likelihood that the estimated tipping point is statistically significant instead of chance alone).

We formalize the above tasks in the Bayesian setting. Recall from Table 2 of §4, that  $\theta_{a,Ag}^E$  (respectively,  $\theta_{a,DisAg}^E$ ) are the estimates of agreeing and disagreeing nature of an author and  $\theta_{a,Ag}^E + \theta_{a,DisAg}^E = 1$ . Let  $TP(\tau)$  denote the event that *in*

*expectation* a threshold value of  $0 < \tau < 1$  serves as a tipping point of disagreement beyond which intolerance is exhibited. Note that we emphasize the term “in expectation” (taken over all authors). We do not mean that every author whose disagreement,  $\theta_{a,DisAg}^E > \tau$ , is intolerant. The empirical likelihood of  $TP(\tau)$  can be expressed by the following probability expression:

$$\mathcal{L}(TP(\tau)) = E[P(\theta_{a,DisAg}^E > \tau | a = I) - P(\theta_{a,DisAg}^E > \tau | a = T)] \quad (8)$$

The events  $a = I$  and  $a = T$  denote that author  $a$  is intolerant and tolerant respectively. The expectation is taken over authors. Showing that  $\tau$  indeed serves as the tipping point of disagreement to exhibit intolerance corresponds to rejecting the null hypothesis that the probabilities in (8) are equal. We employ a Fisher’s exact test to test significance and report confidence measures (using  $p$ -values) for the tipping point thresholds. The results are shown in Table 6.

The threshold  $\tau$  is computed using the entropy method in (Fayyad and Irani, 1993) as follows: We first fit our previously learned model (using the data in Table 1 (a)) to the new threads in Table 6 and its users and posts to obtain the estimates of  $\theta_{a,DisAg}^E$  and other latent variables for feature generation. The learned classifier in §6.1 is used to predict the nature of users (tolerant vs.

intolerant) in the new threads<sup>7</sup>. Then, for each user we have his predicted deliberative (social) psyche (Tolerant vs. Intolerant) and also his overall disagreeing nature exhibited in that thread (the posterior on  $\theta_{a,DisAg}^E \in [0, 1]$ ). For a thread, tolerant and intolerant users (data points) span the range [0, 1] attaining different values for  $\theta_{a,DisAg}^E$ . Each candidate tipping point of disagreement,  $0 \leq \tau' \leq 1$  results in a binary partition of the range with each partition containing some proportion of tolerant and intolerant users. We compute the entropy of the partition for every candidate tipping point in the range [0, 1]. The final tipping point threshold,  $\tau$  is chosen such that it minimizes the partition entropy based on the binary cut-point method in (Fayyad and Irani, 1993).

Since we perform a thread level analysis, the results in Table 6 are thread/issue specific. We note the following from Table 6:

1. Across all threads/issues, we find that the expected disagreement over all posts,  $d$ ,  $E[\theta_{a,d,DisAg}^E] > 0.5$  showing that in discussions of the reported issues, disagreement predominates.
2.  $E[\theta_{a,d,DisAg}^E]$  also gives an estimate of overall *heat* in the issue being discussed. We find sensitive issues like *abortion* and *socialism* being more heated than *healthcare*, *Obama*, etc.
3. The percentage of intolerant users increases with the expected overall disagreement in the issue except for the issue *Obama euphoria*.
4. The estimated tipping point of disagreement to exhibit intolerance,  $\tau$  happens to vary inversely with the expected disagreement,  $E[\theta_{a,d,DisAg}^E]$  except the issue *Obama euphoria*. This reflects that as overall disagreement in the issue increases, the tipping point of intolerance decreases, i.e., due to high discussion heat, people are likely to turn intolerant even with relatively small amount of disagreement. This finding dovetails with prior studies in psychology (Rokeach and Fruchter, 1956) that heated discussions are likely to reduce thresh-

<sup>7</sup> Although this prediction may not be perfect, it can be regarded as considerably reliable to study the trend of tolerance across different issues as our classifier (in §6.1) attains a high (83%) classification accuracy using the full feature set. As judging all users across all threads would require reading about 7000 posts, for confirmation, we randomly sampled 30 authors across various threads for labeling by our judges. 28 out of 30 predictions produced by the classifier correlated with the judges' labels, which should be sufficiently accurate for our analysis.

olds of reception leading to dogmatism, egotism, and intolerance. Table 6 shows that for moderately heated issues (*healthcare*, *Europe's collapse*), in expectation, author's disagreement  $\theta_{a,DisAg}^E$  should exceed 61-65% to exhibit intolerance. However, for sensitive issues, we find that the tipping point is much lower, *abortion*: 37%; *socialism*: 48%.

5. The issue *Obama Euphoria* is an exception to other issues' trends. Even though in expectation, it has  $E[\theta_{a,d,DisAg}^E] = 66\%$  overall disagreement, the percentage of intolerant users remains the lowest (30%) and the tipping point attains a highest value ( $\tau = 0.71$ ), showing more tolerance on the issue. A plausible reason could be that Obama is somewhat more liked and hence attracts less intolerance from users<sup>8</sup>.
6. The  $p$ -values of the estimated tipping points,  $\tau$  across all issues are statistically significant at 98-99% confidence levels.

## 7 Conclusion

This work performed a deep analysis of the sociopsychological and psycholinguistic phenomenon of tolerance in online discussions, which is an important concept in the field of communications. A novel framework is proposed, which is capable of characterizing and classifying tolerance in online discussions. Further, a novel technique was also proposed to quantitatively evaluate the interplay of tolerance and disagreement. Our empirical results using real-life online discussions render key insights into the psycholinguistic process of tolerance and dovetail with existing theories in psychology and communications. To the best of our knowledge, this is the first such quantitative study. In our future work, we want to further this research and study the role of diversity of opinions in the context of tolerance and its relation to polarization.

## Acknowledgments

This work was supported in part by a grant from National Science Foundation (NSF) under grant no. IIS-1111092.

<sup>8</sup> This observation may be linked to the political phenomenon of "democratic citizenship through exposure to diverse perspectives" (Mutz, 2006) where it was shown that exposure to heterogeneous opinions (i.e., greater disagreement), often enhances tolerance.

## References

- Abu-Jbara, A., Dasigi, P., Diab, M. and Dragomir Radev. 2012. Subgroup detection in ideological discussions. *ACL*.
- Agrawal, R. Rajagopalan, S. Srikant, R. Xu. Y. 2003. Mining newsgroups using networks arising from social behavior. *WWW*.
- Bansal, M., Cardie, C., and Lee, L. 2008. The power of negative thinking: Exploiting label disagreement in the min-cut classification framework. In *COLING*.
- Blei, D., A. Ng, and M. Jordan. 2003. Latent Dirichlet Allocation. In *JMLR*.
- Boyer, K.; Grafsgaard, J.; Ha, E. Y.; Phillips, R.; and Lester, J. 2011. An affect-enriched dialogue act classification model for task-oriented dialogue. In *ACL*.
- Burfoot, C., S. Bird, and T. Baldwin. 2011. Collective Classification of Congressional Floor-Debate Transcripts. In *ACL*.
- Cappella, J. N., Price, V., and Nir, L. 2002. Argument repertoire as a reliable and valid measure of opinion quality: electronic dialogue during campaign 2000. *Political Communication. Political Communication*.
- Chen, Z., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M., Ghosh, R. 2013. Leveraging Multi-Domain Prior Knowledge in Topic Models. In *IJCAI*.
- Chung, C. K., and Pennebaker, J. W. 2007. Revealing people's thinking in natural language: Using an automated meaning extraction method in open-ended self-descriptions. *J. of Research in Personality*.
- Choi, Y. and Cardie, C. 2010. Hierarchical sequential learning for extracting opinions and their attributes. In *ACL*.
- Critchley, M. 1964. The neurology of psychotic speech. *The British Journal of Psychiatry*.
- Crocker, D. A. 2005. Tolerance and Deliberative Democracy. *UMD Technical Report*.
- Dahlgren, P. 2002. In search of the talkative public: Media, deliberative democracy and civic culture. *Javnost/The Public*.
- Dahlgren, Peter. 2005. The Internet, Public Spheres, and Political Communication: Dispersion and Deliberation. *Political Communication*.
- Escobar, O. 2012. Public Dialogue and Deliberation: A communication perspective for publicengagement practitioners. *Handbook and Technical Report*.
- Fayyad, U., and Irani, K. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In *UAI*.
- Fishkin, J. 1991. Democracy and deliberation. *New Haven, CT: Yale University Press*.
- Flor, M., and Hadar, U. 2005. The production of metaphoric expressions in spontaneous speech: A controlled-setting experiment. *Metaphor and Symbol*.
- Galley, M., K. McKeown, J. Hirschberg, E. Shriberg. 2004. Identifying agreement and disagreement in conversational speech: Use of Bayesian networks to model pragmatic dependencies. In *ACL*.
- Gastil, J. 2005. Communication as Deliberation: A Non-Deliberative Polemic on Communication Theory. *Univ. of Washington, Technical Report*.
- Gastil, J., and Dillard, J. P. 1999. Increasing political sophistication through public deliberation. *Political Communication*.
- Gastil, John. 2007. Political communication and deliberation. *Sage Publications*.
- Griffiths, T. and Steyvers, M. 2004. Finding scientific topics. In *PNAS*.
- Gutmann, A., and Thompson, D. F. 1996. *Democracy and disagreement*. Harvard University Press.
- Habermas. 1984. The theory of communicative action: Reason and rationalization of society. (*T. McCarthy, Trans. Vol. 1*). Boston, MA: Beacon Press.
- Hillard, D., Ostendorf, M., and Shriberg, E. 2003. Detection of Agreement vs. Disagreement in Meetings: Training with Unlabeled Data. *HLT-NAACL*.
- Hansen, G. J., and Hyunjung, K. 2011. Is the media biased against me? A meta-analysis of the hostile media effect research. *Communication Research Reports*, 28, 169-179.
- Hassan, A. and Radev, D. 2010. Identifying text polarity using random walks. In *ACL*.
- Hofmann, T. 1999. Probabilistic latent semantic analysis. In *UAI*.
- Hu, M. and Liu, B. 2004. Mining and summarizing customer reviews. In *SIGKDD*.
- Joachims, T. Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.
- Kim, S. and Hovy, E. 2007. Crystal: Analyzing predictive opinions on the web. In *EMNLP-CoNLL*.
- Landis, J. R. and Koch, G. G. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 159-174.
- Lin, W. H., and Hauptmann, A. 2006. Are these documents written from different perspectives?: a test of different perspectives based on statistical distribution divergence. In *ACL*.
- Liu, B. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publisher, USA.
- Luskin, R. C., Fishkin, J. S., and Iyengar, S. 2004. Considered Opinions on U.S. Foreign Policy: Face-to-Face versus Online Deliberative Polling. *International Communication Association, New Orleans, LA*.
- Mayfield, E. and Rose, C. P. 2011. Recognizing Authority in Dialogue with an Integer Linear Programming Constrained Model. In *ACL*.
- Moxey, L. M., and Sanford, A. J. 2000. Communicating quantities: A review of psycholinguistic evidence of how expressions determine perspectives. *Applied Cognitive Psychology*.
- Morbini, F. and Sagae, K. 2011. Joint Identification and Segmentation of Domain-Specific Dialogue Acts for Conversational Dialogue Systems. In *ACL*.

- Murakami, A., and Raymond, R. 2010. Support or Oppose? Classifying Positions in Online Debates from Reply Activities and Opinion Expressions. In *COLING*.
- Mukherjee, A. and Liu, B. 2013. Discovering User Interactions in Ideological Discussions. In *ACL*.
- Mukherjee, A. and Liu, B. 2012a. Mining Contentions from Discussions and Debates. In *KDD*.
- Mukherjee, A. and Liu, B. 2012b. Modeling review Comments. In *ACL*.
- Mukherjee, A. and Liu, B. 2012c. Aspect Extraction through Semi-Supervised Modeling. In *ACL*.
- Mukherjee, A. and Liu, B. 2012d. Analysis of Linguistic Style Accommodation in Online Debates. In *COLING*.
- Mutz, D. 2006. *Hearing the Other Side: Deliberative Versus Participatory Democracy*. Cambridge: Cambridge University Press, 2006.
- Pang, B. and Lee, L. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*.
- Pennebaker, J. W., Chung, C. K., Ireland, M., Gonzales, A., and Booth, R. J. 2007. The development and psychometric properties of LIWC2007. *LIWC.Net*.
- Popescu, A. and Etzioni, O. 2005. Extracting product features and opinions from reviews. In *EMNLP*.
- Price, V., Cappella, J. N., and Nir, L. 2002. Does disagreement contribute to more deliberative opinion? *Political Communication*.
- Rokeach, M., and Fruchter, B. 1956. A factorial study of dogmatism and related concepts. *The Journal of Abnormal and Social Psychology*.
- Ryfe, D. M. (2005). Does deliberative democracy work? *Annual review of political science*.
- Slavin, M. O., and Kriegman, D. 1992. *The adaptive design of the human psyche: Psychoanalysis, evolutionary biology, and the therapeutic process*. Guilford Press.
- Somasundaran, S., J. Wiebe. 2009. Recognizing stances in online debates. In *ACL-IJCNLP*.
- Stromer-Galley, J. 2005. Conceptualizing and Measuring Coherence in Online Chat. *Annual Meeting of the International Communication Association*.
- Sunstein, C. R. 2002. The law of group polarization. *Journal of political philosophy*.
- Thomas, M., B. Pang and L. Lee. 2006. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *EMNLP*.
- Wang, L., Lui, M., Kim, S. N., Nivre, J., and Baldwin, T. 2011. Predicting thread discourse structure over technical web forums. In *EMNLP*.
- Wiebe, J. 2000. Learning subjective adjectives from corpora. In *Proc. of National Conference on AI*.
- Yessenalina, A., Yue, A., Cardie, C. 2010. Multi-level structured models for document-level sentiment classification. In *EMNLP*.
- Zhao, X., J. Jiang, H. Yan, and X. Li. 2010. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In *EMNLP*.
- Zingo, M. T. (1998). *Sex/gender Outsiders, Hate Speech, and Freedom of Expression: Can They Say that about Me?* Praeger Publishers.

# Offspring from Reproduction Problems: What Replication Failure Teaches Us

**Antske Fokkens and Marieke van Erp**

The Network Institute

VU University Amsterdam

Amsterdam, The Netherlands

{a.s.fokkens,m.g.j.van.erp}@vu.nl

**Marten Postma**

Utrecht University

Utrecht, The Netherlands

martenp@gmail.com

**Ted Pedersen**

Dept. of Computer Science

University of Minnesota

Duluth, MN 55812 USA

tpederse@d.umn.edu

**Piek Vossen**

The Network Institute

VU University Amsterdam

Amsterdam, The Netherlands

piek.vossen@vu.nl

**Nuno Freire**

The European Library

The Hague, The Netherlands

nfreire@gmail.com

## Abstract

Repeating experiments is an important instrument in the scientific toolbox to validate previous work and build upon existing work. We present two concrete use cases involving key techniques in the NLP domain for which we show that reproducing results is still difficult. We show that the deviation that can be found in reproduction efforts leads to questions about how our results should be interpreted. Moreover, investigating these deviations provides new insights and a deeper understanding of the examined techniques. We identify five aspects that can influence the outcomes of experiments that are typically not addressed in research papers. Our use cases show that these aspects may change the answer to research questions leading us to conclude that more care should be taken in interpreting our results and more research involving systematic testing of methods is required in our field.

## 1 Introduction

Research is a collaborative effort to increase knowledge. While it includes validating previous approaches, our experience is that most research output in our field focuses on presenting new approaches, and to a somewhat lesser extent building upon existing work.

In this paper, we argue that the value of research that attempts to replicate previous approaches goes beyond simply *validating* what is already known. It is also an essential aspect for *building upon* existing approaches. Especially when validation

fails or variations in results are found, systematic testing helps to obtain a clearer picture of both the approach itself and of the meaning of state-of-the-art results leading to a *better insight* into the quality of new approaches in relation to previous work.

We support our claims by presenting two use cases that aim to reproduce results of previous work in two key NLP technologies: measuring WordNet similarity and Named Entity Recognition (NER). Besides highlighting the difficulty of repeating other researchers' work, new insights about the approaches emerged that were not presented in the original papers. This last point shows that reproducing results is not merely part of good practice in science, but also an essential part in gaining a better understanding of the methods we use. Likewise, the problems we face in reproducing previous results are not merely frustrating inconveniences, but also pointers to research questions that deserve deeper investigation.

We investigated five aspects that cause experimental variation that are not typically described in publications: **preprocessing** (e.g. tokenisation), **experimental setup** (e.g. splitting data for cross-validation), **versioning** (e.g. which version of WordNet), **system output** (e.g. the exact features used for individual tokens in NER), and **system variation** (e.g. treatment of ties).

As such, reproduction provides a platform for systematically testing individual aspects of an approach that contribute to a given result. What is the influence of the size of the dataset, for example? How does using a different dataset affect the results? What is a reasonable divergence between different runs of the same experiment? Finding answers to these questions enables us to better interpret our state-of-the-art results.

Moreover, the experiments in this paper show that even while strictly trying to replicate a previous experiment, results may vary up to a point where they lead to different answers to the main question addressed by the experiment. The WordNet similarity experiment use case compares the performance of different similarity measures. We will show that the answer as to which measure works best changes depending on factors such as the gold standard used, the strategy towards part-of-speech or the ranking coefficient, all aspects that are typically not addressed in the literature.

The main contributions of this paper are the following:

- 1) An in-depth analysis of two reproduction use cases in NLP
- 2) New insights into the state-of-the-art results for WordNet similarities and NER, found because of problems in reproducing prior research
- 3) A categorisation of aspects influencing reproduction of experiments and suggestions on testing their influence systematically

The code, data and experimental setup for the WordNet experiments are available at <http://github.com/antske/WordNetSimilarity>, and for the NER experiments at <http://github.com/Mvanerp/NER>. The experiments presented in this paper have been repeated by colleagues not involved in the development of the software using the code included in these repositories. The remainder of this paper is structured as follows. In Section 2, previous work is discussed. Sections 3 and 4 describe our real-world use cases. In Section 5, we present our observations, followed by a more general discussion in Section 6. In Section 7, we present our conclusions.

## 2 Background

This section provides a brief overview of recent work addressing reproduction and benchmark results in computer science related studies and discusses how our research fits in the overall picture.

Most researchers agree that validating results entails that a method should lead to the same overall conclusions rather than producing the exact same numbers (Drummond, 2009; Dalle, 2012; Buchert and Nussbaum, 2012, etc.). In other words, we should strive to *reproduce* the same answer to a research question by different means,

perhaps by re-implementing an algorithm or evaluating it on a new (in domain) data set. *Replication* has a somewhat more limited aim, and simply involves running the exact same system under the same conditions in order to get the exact same results as output.

According to Drummond (2009) replication is not interesting, since it does not lead to new insights. On this point we disagree with Drummond (2009) as replication allows us to: 1) validate prior research, 2) improve on prior research without having to rebuild software from scratch, and 3) compare results of reimplementations and obtain the necessary insights to perform reproduction experiments. The outcome of our use cases confirms the statement that deeper insights into an approach can be obtained when all resources are available, an observation also made by Ince et al. (2012).

Even if exact replication is not a goal many strive for, Ince et al. (2012) argue that insightful reproduction can be an (almost) impossible undertaking without the source code being available. Moreover, it is not always clear where replication stops and reproduction begins. Dalle (2012) distinguishes levels of reproducing results related to how close they are to the original work and how each contributes to research. In general, an increasing awareness of the importance of reproduction research and open code and data can be observed based on publications in high-profile journals (e.g. Nature (Ince et al., 2012)) and initiatives such as myExperiment.<sup>1</sup>

Howison and Herbsleb (2013) point out that, even though this is important, often not enough (academic) credit is gained from making resources available. What is worse, the same holds for research that investigates existing methods rather than introducing new ones, as illustrated by the question that is found on many review forms ‘how novel is the presented approach?’. On the other hand, initiatives for journals addressing exactly this issue (Neylon et al., 2012) and tracks focusing on results verification at conferences such as VLDB<sup>2</sup> show that this opinion is not universal.

A handful of use cases on reproducing or replicating results have been published. Louridas and Gousios (2012) present a use case revealing that source code alone is not enough for reproducing

<sup>1</sup><http://www.myexperiment.org>

<sup>2</sup><http://www.vldb.org/2013/>

results, a point that is also made by Mende (2010) who provides an overview of all information required to replicate results.

The experiments in this paper provide use cases that confirm the points brought out in the literature mentioned above. This includes both observations that a detailed level of information is required for truly insightful reproduction research as well as the claim that such research leads to better understanding of our techniques. Furthermore, the work in this paper relates to Bikel (2004)'s work. He provides all information needed in addition to Collins (1999) to replicate Collins' benchmark results. Our work is similar in that we also aim to fill in the blanks needed to replicate results. It must be noted, however, that the use cases in this paper have a significantly smaller scale than Bikel's.

Our research distinguishes itself from previous work, because it links the challenges of reproduction to what they mean for reported results beyond validation. Ruml (2010) mentions variations in outcome as a reason not to emphasise comparisons to benchmarks. Vanschoren et al. (2012) propose to use experimental databases to systematically test variations for machine learning, but neither links the two issues together. Raeder et al. (2010) come closest to our work in a critical study on the evaluation of machine learning. They show that choices in the methodology, such as data sets, evaluation metrics and type of cross-validation can influence the conclusions of an experiment, as we also find in our second use case. However, they focus on the problem of evaluation and recommendations on how to achieve consistent reproducible results. Our contribution is to investigate how much results vary. We cannot control how fellow researchers carry out their evaluation, but if we have an idea of the variations that typically occur within a system, we can better compare approaches for which not all details are known.

### 3 WordNet Similarity Measures

Patwardhan and Pedersen (2006) and Pedersen (2010) present studies where the output of a variety of WordNet similarity and relatedness measures are compared. They rank Miller and Charles (1991)'s set (henceforth "*mc-set*") of 30 word pairs according to their semantic relatedness with several WordNet similarity measures.

Each measure ranks the *mc-set* of word pairs and these outputs are compared to Miller and

Charles (1991)'s gold standard based on human rankings using the Spearman's Correlation Coefficient (Spearman, 1904,  $\rho$ ). Pedersen (2010) also ranks the original set of 65 word pairs ranked by humans in an experiment by Rubenstein and Goodenough (1965) (*rg-set*) which is a superset of Miller and Charles's set.

#### 3.1 Replication Attempts

This research emerged from a project running a similar experiment for Dutch on Cornetto (Vossen et al., 2013). First, an attempt was made to reproduce the results reported in Patwardhan and Pedersen (2006) and Pedersen (2010) on the English WordNet using their WordNet::Similarity web-interface.<sup>3</sup> Results differed from those reported in the aforementioned works, even when using the same versions as the original, WordNet::Similarity-1.02 and WordNet 2.1 (Patwardhan and Pedersen, 2006) and WordNet::Similarity-2.05 and WordNet 3.0 (Pedersen, 2010), respectively.<sup>4</sup>

The fact that results of similarity measures on WordNet can differ even while the same software and same versions are used indicates that properties which are not addressed in the literature may influence the output of similarity measures. We therefore conducted a range of experiments that, in addition to searching for the right settings to replicate results of previous research, address the following questions:

- 1) Which properties have an impact on the performance of WordNet similarity measures?
- 2) How much does the performance of individual measures vary?
- 3) How do commonly used measures compare when the variation of their performance are taken into account?

#### 3.2 Methodology and first observations

The questions above were addressed in two stages. In the first stage, Fokkens, who was not involved in the first replication attempt implemented a script to calculate similarity measures using WordNet::Similarity. This included similarity measures introduced by Wu and Palmer (1994) (*wup*),

<sup>3</sup>Obtained from <http://talisker.d.umn.edu/cgi-bin/similarity/similarity.cgi>, WordNet::Similarity version 2.05. This web interface has now moved to <http://maraca.d.umn.edu>

<sup>4</sup>WordNet::Similarity were obtained <http://search.cpan.org/dist/WordNet-Similarity/>.

Leacock and Chodorow (1998) (*lch*), Resnik (1995) (*res*), Jiang and Conrath (1997) (*jcn*), Lin (1998) (*lin*), Banerjee and Pedersen (2003) (*lesk*), Hirst and St-Onge (1998) (*hso*) and Patwardhan and Pedersen (2006) (*vector* and *vpairs*) respectively.

Consequently, settings and properties were changed systematically and shared with Pedersen who attempted to produce the new results with his own implementations. First, we made sure that the script implemented by Fokkens could produce the same WordNet similarity scores for each individual word pair as those used to calculate the ranking on the *mc-set* by Pedersen (2010). Finally, the gold standard and exact implementation of the Spearman ranking coefficient were compared.

Differences in results turned out to be related to variations in the **experimental setup**. First, we made different assumptions on the restriction of part-of-speech tags (henceforth “PoS-tag”) considered in the comparison. Miller and Charles (1991) do not discuss how they deal with words with more than one PoS-tag in their study. Pedersen therefore included all senses with any PoS-tag in his study. The first replication attempt had restricted PoS-tags to nouns based on the idea that most items are nouns and subjects would be primed to primarily think of the noun senses. Both assumptions are reasonable. Pos-tags were not restricted in the second replication attempt, but because of a bug in the code only the first identified PoS-tag (“noun” in all cases) was considered. We therefore mistakenly assumed that PoS-tag restrictions did not matter until we compared individual scores between Pedersen and the replication attempts.

Second, there are two gold standards for the Miller and Charles (1991) set: one has the scores assigned during the original experiment run by Rubenstein and Goodenough (1965), the other has the scores assigned during Miller and Charles (1991)’s own experiment. The ranking correlation between the two sets is high, but they are not identical. Again, there is no reason why one gold standard would be a better choice than the other, but in order to replicate results, it must be known which of the two was used. Third, results changed because of differences in the treatment of ties while calculating Spearman  $\rho$ . The influence of the exact gold standard and calculation of Spearman  $\rho$  could only be found because Pedersen could pro-

measure	Spearman $\rho$		Kendall $\tau$		ranking variation
	min	max	min	max	
path based similarity					
path	0.70	0.78	0.55	0.62	1-8
wup	0.70	0.79	0.53	0.61	1-6
lch	0.70	0.78	0.55	0.62	1-7
path based information content					
res	0.65	0.75	0.26	0.57	4-11
lin	0.49	0.73	0.36	0.53	6-10
jcn	0.46	0.73	0.32	0.55	5, 7-11
path based relatedness					
hso	0.73	0.80	0.36	0.41	1-3,5-10
dictionary and corpus based relatedness					
vpairs	0.40	0.70	0.26	0.50	7-11
vector	0.48	0.92	0.33	0.76	1,2,4,6-11
lesk	0.66	0.83	-0.02	0.61	1-8,11,12

Table 1: Variation WordNet measures’ results

vide the output of the similarity measures he used to calculate the coefficient. It is unlikely we would have been able to replicate his results at all without the output of this intermediate step. Finally, results for *lch*, *lesk* and *wup* changed according to measure specific configuration settings such as including a PoS-tag specific root node or turning on normalisation.

In the second stage of this research, we ran experiments that systematically manipulate the influential factors described above. In this experiment, we included both the *mc-set* and the complete *rg-set*. The implementation of Spearman  $\rho$  used in Pedersen (2010) assigned the lowest number in ranking to ties rather than the mean, resulting in an unjustified drop in results for scores that lead to many ties. We therefore experimented with a different correlation measure, Kendall tau coefficient (Kendall, 1938,  $\tau$ ) rather than two versions of Spearman  $\rho$ .

### 3.3 Variation per measure

All measures varied in their performance. The complete outcome of our experiments (both the similarity measures assigned to each pair as well as the output of the ranking coefficients) are included in the data set provided at <http://github.com/antske/WordNetSimilarity>. Table 1 presents an overview of the main point we wish to make through this experiment: the minimal and maximal results according to both ranking coefficients. Results for similarity measures varied from 0.06-0.42 points for Spearman  $\rho$  and from 0.05-0.60 points for Kendall  $\tau$ . The last column indicates the variation of performance of a measure

compared to the other measures, where 1 is the best performing measure and 12 is the worst.<sup>5</sup> For instance, `path` has been best performing measure, second best, eighth best and all positions in between, `vector` has ranked first, second and fourth, but also occupied all positions from six to eleven.

In principle, it is to be expected that numbers are not exactly the same while evaluating against a different data set (the *mc-set* versus the *rg-set*), taking a different set of synsets to evaluate on (changing PoS-tag restrictions) or changing configuration settings that influence the similarity score. However, a variation of up to 0.44 points in Spearman  $\rho$  and 0.60 in Kendall  $\tau$ <sup>6</sup> leads to the question of how indicative these results really are. A more serious problem is the fact that the comparative performance of individual measure changes. Which measure performs best depends on the evaluation set, ranking coefficient, PoS-tag restrictions and configuration settings. This means that the answer to the question of which similarity measure is best to mimic human similarity scores depends on aspects that are often not even mentioned, let alone systematically compared.

### 3.4 Variation per category

For each influential category of experimental variation, we compared the variation in Spearman  $\rho$  and Kendall  $\tau$ , while similarity measure and other influential categories were kept stable. The categories we varied include WordNet and WordNet::Similarity version, the gold standard used to evaluate, restrictions on PoS-tags, and measure specific configurations. Table 2 presents the maximum variation found across measures for each category. The last column indicates how often the ranking of a specific measure changed as the category changed, e.g. did the measure ranking third using specific configurations, PoS-tag restrictions and a specific gold standard using WordNet 2.1 still rank third when WordNet 3.0 was used instead? The number in parentheses next to the ‘different ranks’ in the table presents the total number of scores investigated. Note that this number changes for each category, because we com-

<sup>5</sup>Some measures ranked differently as their individual configuration settings changed. In these cases, the measure was included in the overall ranking multiple times, which is why there are more ranking positions than measures.

<sup>6</sup>Section 3.4 explains why the variation in Kendall is this extreme and  $\rho$  is more appropriate for this task.

Variation	Maximum difference		Different rank (tot)
	Spearman $\rho$	Kendall $\tau$	
WN version	0.44	0.42	223 (252)
gold standard	0.24	0.21	359 (504)
PoS-tag	0.09	0.08	208 (504)
configuration	0.08	0.60	37 (90)

Table 2: Variations per category

pared two WordNet versions (WN version), three gold standard and PoS-tag restriction variations and configuration only for the subset of scores where configuration matters.

There are no definite statements to make as to which version (Patwardhan and Pedersen (2006) vs Pedersen (2010)), PoS-tag restriction or configuration gives the best results. Likewise, while most measures do better on the smaller data set, some achieve their highest results on the full set. This is partially due to the fact that ranking coefficients are sensitive to outliers. In several cases where PoS-tag restrictions led to different results, only one pair received a different score. For instance, `path` assigns a relatively high score to the pair *chord-smile* when verbs are included, because the hierarchy of verbs in WordNet is relatively flat. This effect is not observed in `wup` and `lch` which correct for the depth of the hierarchy. On the other hand, `res`, `lin` and `jcn` score better on the same set when verbs are considered, because they cannot detect any relatedness for the pair *crane-implement* when restricted to nouns.

On top of the variations presented above, we notice a discrepancy between the two coefficients. Kendall  $\tau$  generally leads to lower coefficient scores than Spearman  $\rho$ . Moreover, they each give different relative indications: where `lesk` achieves its highest Spearman  $\rho$ , it has an extremely low Kendall  $\tau$  of 0.01. Spearman  $\rho$  uses the difference in rank as its basis to calculate a correlation, where Kendall  $\tau$  uses the number of items with the correct rank. The low Kendall  $\tau$  for `lesk` is the result of three pairs receiving a score that is too high. Other pairs that get a relatively accurate score are pushed one place down in rank. Because only items that receive the exact same rank help to increase  $\tau$ , such a shift can result in a drastic drop in the coefficient. In our opinion, Spearman  $\rho$  is therefore preferable over Kendall  $\tau$ . We included  $\tau$ , because many authors do not mention the ranking coefficient they use (cf. Budanitsky and Hirst (2006), Resnik (1995)) and both  $\rho$  and  $\tau$  are com-

monly used coefficients.

Except for WordNet, which Budanitsky and Hirst (2006) hold accountable for minor variations in a footnote, the influential categories we investigated in this paper, to our knowledge, have not yet been addressed in the literature. Cramer (2008) points out that results from WordNet-Human similarity correlations lead to scattered results reporting variations similar to ours, but she compares studies using different measures, data and experimental setup. This study shows that even if the main properties are kept stable, results vary enough to change the identity of the measure that yields the best performance. Table 1 reveals a wide variation in ranking relative to alternative approaches. Results in Table 2 show that it is common for the ranking of a score to change due to variations that are not at the core of the method.

This study shows that it is far from clear how different WordNet similarity measures relate to each other. In fact, we do not know how we can obtain the best results. This is particularly challenging, because the ‘best results’ may depend on the intended use of the similarity scores (Meng et al., 2013). This is also the reason why we presented the maximum variation observed, rather than the average or typical variation (mostly below 0.10 points). The experiments presented in this paper resulted in a vast amount of data. An elaborate analysis of this data is needed to get a better understanding of how measures work and why results vary to such an extent. We leave this investigation to future work. If there is one take-home message from this experiment, it is that one should experiment with parameters such as restrictions on PoS-tags or configurations and determine which score to use depending on what it is used for, rather than picking something that did best in a study using different data for a different task and may have used a different version of WordNet.

## 4 Reproducing a NER method

Freire et al. (2012) describe an approach to classifying named entities in the cultural heritage domain. The approach is based on the assumption that domain knowledge, encoded in complex features, can aid a machine learning algorithm in NER tasks when only little training data is available. These features include information about person and organisation names, locations, as well as PoS-tags. Additionally, some general features

are used such as a window of three preceding and two following tokens, token length and capitalisation information. Experiments are run in a 10-fold cross-validation setup using an open source machine learning toolkit (McCallum, 2002).

### 4.1 Reproducing NER Experiments

This experiment can be seen as a real-world case of *the sad tale of the Ziggiebottom tagger* (Pedersen, 2008). The (fictional) Ziggiebottom tagger is a tagger with spectacular results that looks like it will solve some major problems in your system. However, the code is not available and a new implementation does not yield the same results. The original authors cannot provide the necessary details to reproduce their results, because most of the work has been done by a PhD student who has finished and moved on to something else. In the end, the newly implemented Ziggiebottom tagger is not used, because it does not lead to the promised better results and all effort went to waste.

Van Erp was interested in the NER approach presented in Freire et al. (2012). Unfortunately, the code could not be made available, so she decided to reimplement the approach. Despite feedback from Freire about particular details of the system, results remained 20 points below those reported in Freire et al. (2012) in overall F-score (Van Erp and Van der Meij, 2013).

The reimplementing process involved choices about seemingly small details such as rounding to how many decimals, how to tokenise or how much data cleanup to perform (normalisation of non-alphanumeric characters for example). Trying different parameter combinations for feature generation and the algorithm never yielded the exact same results as Freire et al. (2012). The results of the best run in our first reproduction attempt, together with the original results from Freire et al. (2012) are presented in Table 3. Van Erp and Van der Meij (2013) provide an overview of the implementation efforts.

### 4.2 Following up from reproduction

Since the experiments in Van Erp and Van der Meij (2013) introduce several new research questions regarding the influence of data cleaning and the limitations of the dataset, we performed some additional experiments.

First, we varied the tokenisation, removing non-alphanumeric characters from the data set. This yielded a significantly smaller data set (10,442

	(Freire et al., 2012) results			Van Erp and Van der Meij’s replication results		
	Precision	Recall	$F_{\beta=1}$	Precision	Recall	$F_{\beta=1}$
LOC (388)	92%	55%	69	77.80%	39.18%	52.05
ORG (157)	90%	57%	70	65.75%	30.57%	41.74
PER (614)	91%	56%	69	73.33%	37.62%	49.73
Overall (1,159)	91%	55%	69	73.33%	37.19%	49.45

Table 3: Precision, recall and  $F_{\beta=1}$  scores for the original experiments from Freire et al. 2012 and our replication of their approach as presented in Van Erp and Van der Meij (2013)

tokens vs 12,510), and a 15 point drop in overall F-score. Then, we investigated whether variation in the cross-validation splits made any difference as we noticed that some NEs were only present in particular fields in the data, which can have a significant impact on a small dataset. We inspected the difference between different cross-validation folds by computing the standard deviations of the scores and found deviations of up to 25 points in F-score between the 10 splits. In the general setup, database records were randomly distributed over the folds and cut off to balance the fold sizes. In a different approach to dividing the data by distributing individual sentences from the records over the folds, performance increases by 8.57 points in overall F-score to 58.02. This is not what was done in the original Freire et al. (2012) paper, but shows that the results obtained with this dataset are quite fragile.

As we worried about the complexity of the feature set relative to the size of the data set, we deviated somewhat from Freire et al. (2012)’s experiments in that we switched some features on and off. Removal of complex features pertaining to the window around the focus token improved our results by 3.84 points in overall F-score to 53.39. The complex features based on VIAF,<sup>7</sup> GeoNames<sup>8</sup> and WordNet do contribute to the classification in the Mallet setup as removing them and only using the focus token, window and generic features causes a slight drop in overall F-score from 49.45 to 47.25.

When training the Stanford NER system (Finkel et al., 2005) on just the tokens from the Freire data set and the parameters from english.all.3class.distsim.prop (included in the Stanford NER release, see also Van Erp and Van der Meij (2013)), our F-scores come very close to those reported by Freire et al. (2012), but mostly with a higher recall and lower precision. It is puzzling that the Stanford system obtains such high

results with only very simple features, whereas for Mallet the complex features show improvement over simpler features. This leads to questions about the differences between the CRF implementations and the influence of their parameters, which we hope to investigate in future work.

### 4.3 Reproduction difficulties explained

Several reasons may be the cause of why we fail to reproduce results. As mentioned, not all resources and data were available for this experiment, thus causing us to navigate in the dark as we could not reverse-engineer intermediate steps, but only compare to the final precision, recall and F-scores.

The experiments follow a general machine learning setup consisting roughly of four steps: preprocess data, generate features, train model and test model. The novelty and replication problems lie in the first three steps. How the data was preprocessed is a major factor here. The data set consisted of XML files marked up with inline named entity tags. In order to generate machine learning features, this data has to be tokenised, possibly cleaned up and the named entity markup had to be converted to a token-based scheme. Each of these steps can be carried out in several ways, and choices made here can have great influence on the rest of the pipeline.

Similar choices have to be made for preprocessing external resources. From the descriptions in the original paper, it is unclear how records in VIAF and GeoNames were preprocessed, or even which versions of these resources were used. Preprocessing and calculating occurrence statistics over VIAF takes 30 hours for each run. It is thus not feasible to identify the main potential variations without the original data to verify this preparatory step.

Numbers had to be rounded when generating the features, leading to the question of how many decimals are required to be discriminative without creating an overly sparse dataset. Freire recalls that encoding features as multi-value discrete fea-

<sup>7</sup><http://www.viaf.org>

<sup>8</sup><http://www.geonames.org>

tures versus several boolean features can have significant impact. These settings are not mentioned in the paper, making reproduction very difficult.

As the project in which the original research was performed has ended, and there is no central repository where such information can be retrieved, we are left to wonder how to reuse this approach in order to further domain-specific NER.

## 5 Observations

In this section, we generalise the observations from our use cases to the main categories that can influence reproduction.

Despite our efforts to describe our systems as clearly as possible, details that can make a tremendous difference are often omitted in papers. It will be no surprise to researchers in the field that **pre-processing** of data can make or break an experiment.

The choice of which steps we perform, and how each of these steps is carried out exactly are part of our **experimental setup**. A major difference in the results for the NER experiments was caused by variations in the way in which we split the data for cross-validation.

As we fine-tune our techniques, software gets updated, data sets are extended or annotation bugs are fixed. In the WordNet experiment, we found that there were two different gold standard data sets. There are also different versions of WordNet, and the WordNet::Similarity packages. Similarly for the NER experiment, GeoNames, VIAF and Mallet are updated regularly. It is therefore critical to pay attention to **versioning**.

Our experiments often consist of several different steps whose outputs may be difficult to retrace. In order to check the output of a reproduction experiment at every step of the way, **system output** of experiments, including intermediate steps, is vital. The WordNet replication was only possible, because Pedersen could provide the similarity scores of each word pair. This enabled us to compare the intermediate output and identify the source of differences in output.

Lastly, there may be inherent **system variations** in the techniques used. Machine learning algorithms may for instance use coin flips in case of a tie. This was not observed in our experiments, but such variations may be determined by running an experiment several times and taking the average over the different runs (cf. Raeder et al. (2010)).

All together, these observations show that sharing data and software play a key role in gaining insight into how our methods work. Vanschoren et al. (2012) propose a setup that allows researchers to provide their full experimental setup, which should include exact steps followed in preprocessing the data, documentation of the experimental setup, exact versions of the software and resources used and experimental output. Having access to such a setup allows other researchers to validate research, but also tweak the approach to investigate system variation, systematically test the approach in order to learn its limitations and strengths and ultimately improve on it.

## 6 Discussion

Many of the aspects addressed in the previous section such as preprocessing are typically only mentioned in passing, or not at all. There is often not enough space to capture all details, and they are generally not the core of the research described. Still, our use cases have shown that they can have a tremendous impact on reproduction, and can even lead to different conclusions. This leads to serious questions on how we can interpret our results and how we can compare the performance of different methods. Is an improvement of a few per cent really due to the novelty of the approach if larger variations are found when the data is split differently? Is a method that does not quite achieve the highest reported state-of-the-art result truly less good? What does a state-of-the-art result mean if it is only tested on one data set?

If one really wants to know whether a result is better or worse than the state-of-the-art, the range of variation within the state-of-the-art must be known. Systematic experiments such as the ones we carried out for WordNet similarity and NER, can help determine this range. For results that fall within the range, it holds that they can only be judged by evaluations going beyond comparing performance numbers, i.e. an evaluation of how the approach achieves a given result and how that relates to alternative approaches.

Naturally, our use cases do not represent the entire gamut of research methodologies and problems in the NLP community. However, they do represent two core technologies and our observations align with previous literature on replication and reproduction.

Despite the systematic variation we employed

in our experiments, they do not answer all questions that the problems in reproduction evoked. For the WordNet experiments, deeper analysis is required to gain full understanding of how individual influential aspects interact with each measurement. For the NER experiments, we are yet to identify the cause of our failure to reproduce.

The considerable time investment required for such experiments forms a challenge. Due to pressure to publish or other time limitations, they cannot be carried out for each evaluation. Therefore, it is important to share our experiments, so that other researchers (or students) can take this up. This could be stimulated by instituting reproduction tracks in conferences, thus rewarding systematic investigation of research approaches. It can also be aided by adopting initiatives that enable authors to easily include data, code and/or workflows with their publications such as the PLOS/figshare collaboration.<sup>9</sup> We already do a similar thing for our research problems by organising challenges or shared tasks, why not extend this to systematic testing of our approaches?

## 7 Conclusion

We have presented two reproduction use cases for the NLP domain. We show that repeating other researchers' experiments can lead to new research questions and provide new insights into and better understanding of the investigated techniques.

Our WordNet experiments show that the performance of similarity measures can be influenced by the PoS-tags considered, measure specific variations, the rank coefficient and the gold standard used for comparison. We not only find that such variations lead to different numbers, but also different rankings of the individual measures, i.e. these aspects lead to a different answer to the question as to which measure performs best. We did not succeed in reproducing the NER results of Freire et al. (2012), showing the complexity of what seems a straightforward reproduction case based on a system description and training data only. Our analyses show that it is still an open question whether additional complex features improve domain specific NER and that this may partially depend on the CRF implementation.

Some observations go beyond our use cases. In particular, the fact that results vary significantly

<sup>9</sup><http://blogs.plos.org/plos/2013/01/easier-access-to-plos-data/>

because of details that are not made explicit in our publications. Systematic testing can provide an indication of this variation. We have classified relevant aspects in five categories occurring across subdisciplines of NLP: **preprocessing, experimental setup, versioning, system output, and system variation.**

We believe that knowing the influence of different aspects in our experimental workflow can help increase our understanding of the robustness of the approach at hand and will help understand the meaning of the state-of-the-art better. Some techniques are reused so often (the papers introducing WordNet similarity measures have around 1,000-2,000 citations each as of February 2013, for example) that knowing their strengths and weaknesses is essential for optimising their use.

As mentioned many times before, sharing is key to facilitating reuse, even if the code is imperfect and contains hacks and possibly bugs. In the end, the same holds for software as for documentation: *it is like sex: if it is good, it is very good and if it is bad, it is better than nothing!*<sup>10</sup> But most of all: when reproduction fails, regardless of whether original code or a reimplementations was used, valuable insights can emerge from investigating the cause of this failure. So don't let your failing reimplementations of the *Zigglebottom* tagger collect dust on a shelf while others reimplement their own failing *Zigglebottoms*. As a community, we need to know where our approaches fail, as much –if not more– as where they succeed.

## Acknowledgments

We would like to thank the anonymous reviewers for their eye to detail and useful comments to make this a better paper. We furthermore thank Ruben Izquierdo, Lourens van der Meij, Christoph Zwirello, Rebecca Dridan and the Semantic Web Group at VU University for their help and useful feedback. The research leading to this paper was supported by the European Union's 7th Framework Programme via the NewsReader Project (ICT-316404), the Agora project, by NWO CATCH programme, grant 640.004.801, and the BiographyNed project, a joint project with Huygens/ING Institute of the Dutch Academy of Sciences funded by the Netherlands eScience Center (<http://esciencecenter.nl/>).

<sup>10</sup>The documentation variant of this quote is attributed to Dick Brandon.

## References

- Stanjeev Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 805–810, Acapulco, August.
- Daniel M. Bikel. 2004. Intricacies of Collins’ parsing model. *Computational Linguistics*, 30(4):479–511.
- Tomasz Buchert and Lucas Nussbaum. 2012. Leveraging business workflows in distributed systems research for the orchestration of reproducible and scalable experiments. In Anne Etien, editor, *9ème édition de la conférence MANifestation des JEunes Chercheurs en Sciences et Technologies de l’Information et de la Communication - MajecSTIC 2012 (2012)*.
- Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.
- Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Phd dissertation, University of Pennsylvania.
- Irene Cramer. 2008. How well do semantic relatedness measures perform? a meta-study. In *Semantics in Text Processing. STEP 2008 Conference Proceedings*, volume 1, pages 59–70.
- Olivier Dalle. 2012. On reproducibility and traceability of simulations. In *WSC-Winter Simulation Conference-2012*.
- Chris Drummond. 2009. Replicability is not reproducibility: nor is it good science. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning: Workshop on Evaluation Methods for Machine Learning IV*.
- Jenny Finkel, Trond Grenager, and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363–370, Ann Arbor, USA.
- Nuno Freire, José Borbinha, and Pável Calado. 2012. An approach for named entity recognition in poorly structured data. In *Proceedings of ESWC 2012*.
- Graeme Hirst and David St-Onge. 1998. Lexical chains as representations of context for the detection and correction of malapropisms. In C. Fellbaum, editor, *WordNet: An electronic lexical database*, pages 305–332. MIT Press.
- James Howison and James D. Herbsleb. 2013. Sharing the spoils: incentives and collaboration in scientific software development. In *Proceedings of the 2013 conference on Computer Supported Cooperative Work*, pages 459–470.
- Darrel C. Ince, Leslie Hatton, and John Graham-Cumming. 2012. The case for open computer programs. *Nature*, 482(7386):485–488.
- Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics (ROCLING X)*, pages 19–33, Taiwan.
- Maurice Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93.
- Claudia Leacock and Martin Chodorow. 1998. Combining local context and WordNet similarity for word sense identification. In C. Fellbaum, editor, *WordNet: An electronic lexical database*, pages 265–283. MIT Press.
- Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, pages 296–304, Madison, USA.
- Panos Louridas and Georgios Gousios. 2012. A note on rigour and replicability. *SIGSOFT Softw. Eng. Notes*, 37(5):1–4.
- Andrew K. McCallum. 2002. MALLET: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- Thilo Mende. 2010. Replication of defect prediction studies: problems, pitfalls and recommendations. In *Proceedings of the 6th International Conference on Predictive Models in Software Engineering*. ACM.
- Lingling Meng, Runqing Huang, and Junzhong Gu. 2013. A review of semantic similarity measures in wordnet. *International Journal of Hybrid Information Technology*, 6(1):1–12.
- George A. Miller and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.
- Cameron Neylon, Jan Aerts, C Titus Brown, Simon J Coles, Les Hatton, Daniel Lemire, K Jarrod Millman, Peter Murray-Rust, Fernando Perez, Neil Saunders, Nigam Shah, Arfon Smith, Gaël Varoquaux, and Egon Willighagen. 2012. Changing computational research. the challenges ahead. *Source Code for Biology and Medicine*, 7(2).
- Siddharth Patwardhan and Ted Pedersen. 2006. Using wordnet based context vectors to estimate the semantic relatedness of concepts. In *Proceedings of the EACL 2006 Workshop Making Sense of Sense - Bringing Computational Linguistics and Psycholinguistics Together*, pages 1–8, Trento, Italy.
- Ted Pedersen. 2008. Empiricism is not a matter of faith. *Computational Linguistics*, 34(3):465–470.

- Ted Pedersen. 2010. Information content measures of semantic similarity perform better without sense-tagged text. In *Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2010)*, pages 329–332, Los Angeles, USA.
- Troy Raeder, T. Ryan Hoens, and Nitesh V. Chawla. 2010. Consequences of variability in classifier performance estimates. In *Proceedings of ICDM'2010*.
- Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 448–453, Montreal, Canada.
- Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.
- Wheeler Ruml. 2010. The logic of benchmarking: A case against state-of-the-art performance. In *Proceedings of the Third Annual Symposium on Combinatorial Search (SOCS-10)*.
- Charles Spearman. 1904. Proof and measurement of association between two things. *American Journal of Psychology*, 15:72—101.
- Marieke Van Erp and Lourens Van der Meij. 2013. Reusable research? a case study in named entity recognition. CLTL 2013-01, Computational Lexicology & Terminology Lab, VU University Amsterdam.
- Joaquin Vanschoren, Hendrik Blockeel, Bernhard Pfahringer, and Geoffrey Holmes. 2012. Experiment databases. *Machine Learning*, 87(2):127–158.
- Piek Vossen, Isa Maks, Roxane Segers, Hennie van der Vliet, Marie-Francine Moens, Katja Hofmann, Erik Tjong Kim Sang, and Maarten de Rijke. 2013. Cornetto: a Combinatorial Lexical Semantic Database for Dutch. In Peter Spyns and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch Results by the STEVIN-programme*, number XVII in Theory and Applications of Natural Language Processing, chapter 10. Springer.
- Zhibiao Wu and Martha Palmer. 1994. Verb semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 133—138, Las Cruces, USA.

# Evaluating Text Segmentation using Boundary Edit Distance

Chris Fournier

University of Ottawa  
Ottawa, ON, Canada

cfour037@eecs.uottawa.ca

## Abstract

This work proposes a new segmentation evaluation metric, named *boundary similarity* (B), an inter-coder agreement coefficient adaptation, and a confusion-matrix for segmentation that are all based upon an adaptation of the boundary edit distance in Fournier and Inkpen (2012). Existing segmentation metrics such as  $P_k$ , WindowDiff, and Segmentation Similarity (S) are all able to award partial credit for near misses between boundaries, but are biased towards segmentations containing few or tightly clustered boundaries. Despite S's improvements, its normalization also produces cosmetically high values that overestimate agreement & performance, leading this work to propose a solution.

## 1 Introduction

Text segmentation is the task of splitting text into segments by placing boundaries within it. Segmentation is performed for a variety of purposes and is often a pre-processing step in a larger task. E.g., text can be topically segmented to aid video and audio retrieval (Franz et al., 2007), question answering (Oh et al., 2007), subjectivity analysis (Stoyanov and Cardie, 2008), and even summarization (Haghighi and Vanderwende, 2009).

A variety of segmentation granularities, or atomic units, exist, including segmentations at the morpheme (e.g., Sirts and Alumäe 2012), word (e.g., Chang et al. 2008), sentence (e.g., Reynar and Ratnaparkhi 1997), and paragraph (e.g., Hearst 1997) levels. Between each atomic unit lies the potential to place a boundary. Segmentations can also represent the structure of text as being organized linearly (e.g., Hearst 1997), hierarchically (e.g., Eisenstein 2009), etc. Theoretically, segmentations could also contain varying bound-

ary types, e.g., two boundary types could differentiate between act and scene breaks in a play.

Because of its value to natural language processing, various text segmentation tasks have been automated such as topical segmentation—for which a variety of automatic segmenters exist (e.g., Hearst 1997, Malioutov and Barzilay 2006, Eisenstein and Barzilay 2008, and Kazantseva and Szpakowicz 2011). This work addresses how to best select an automatic segmenter and which segmentation metrics are most appropriate to do so.

To select an automatic segmenter for a particular task, a variety of segmentation evaluation metrics have been proposed, including  $P_k$  (Beeferman and Berger, 1999, pp. 198–200), WindowDiff (WD; Pevzner and Hearst 2002, p. 10), and most recently Segmentation Similarity (S; Fournier and Inkpen 2012, p. 154–156). Each of these metrics have a variety of flaws:  $P_k$  and WindowDiff both under-penalize errors at the beginning of segmentations (Lamprier et al., 2007) and have a bias towards favouring segmentations with few or tightly-clustered boundaries (Niekrasz and Moore, 2010), while S produces overly optimistic values due to its normalization (shown later).

To overcome the flaws of existing text segmentation metrics, this work proposes a new series of metrics derived from an adaptation of boundary edit distance (Fournier and Inkpen, 2012, p. 154–156). This new metric is named *boundary similarity* (B). A confusion matrix to interpret segmentation as a classification problem is also proposed, allowing for the computation of information retrieval (IR) metrics such as precision and recall.<sup>1</sup>

In this work: §2 reviews existing segmentation metrics; §3 proposes an adaptation of *boundary edit distance*, a new normalization of it, a new confusion matrix for segmentation, and an inter-

<sup>1</sup>An implementation of boundary edit distance, boundary similarity, B-precision, and B-recall, etc. is provided at <http://nlp.chrisfournier.ca/>

coder agreement coefficient adaptation; §4 compares existing segmentation metrics to those proposed herein; §5 evaluates S and B based inter-coder agreement; and §6 compares B, S, and WD while evaluating automatic segmenters.

## 2 Related Work

### 2.1 Segmentation Evaluation

Many early studies evaluated automatic segmenters using information retrieval (IR) metrics such as precision, recall, etc. These metrics looked at segmentation as a binary classification problem and were very harsh in their comparisons—no credit was awarded for nearly missing a boundary.

Near misses occur frequently in segmentation—although manual coders often agree upon the bulk of where segment lie, they frequently disagree upon the exact position of boundaries (Artstein and Poesio, 2008, p. 40). To attempt to overcome this issue, both Passonneau and Litman (1993) and Hearst (1993) conflated multiple manual segmentations into one that contained only those boundaries which the majority of coders agreed upon. IR metrics were then used to compare automatic segmenters to this majority solution. Such a majority solution is unsuitable, however, because it does not contain actual subtopic breaks, but instead the conflation of a collection of potentially disagreeing solutions. Additionally, the definition of what constitutes a majority is subjective (e.g., Passonneau and Litman (1993, p. 150), Litman and Passonneau (1995), Hearst (1993, p. 6) each used  $4/7$ ,  $3/7$ , and  $> 50\%$ , respectively).

To address the issue of awarding partial credit for an automatic segmenter nearly missing a boundary—without conflating segmentations, Beeferman and Berger (1999, pp. 198–200) proposed a new metric named  $P_k$ . Pevzner and Hearst (2002, pp. 3–4) explain  $P_k$  well: a window of size  $k$ —where  $k$  is half of the mean manual segmentation length—is slid across both automatic and manual segmentations. A penalty is awarded if the window’s edges are found to be in differing or the same segments within the manual segmentation and the automatic segmentation disagrees.  $P_k$  is the sum of these penalties over all windows. Measuring the proportion of windows in error allows  $P_k$  to penalize a fully missed boundary by  $k$  windows, whereas a nearly missed boundary is penalized by the distance that it is offset.

$P_k$  was not without issue, however. Pevzner

and Hearst (2002, pp. 5–10) identified that  $P_k$ : i) penalizes false negatives (FNs)<sup>2</sup> more than false positives (FPs); ii) does not penalize full misses within  $k$  units of a reference boundary; iii) penalize near misses too harshly in some situations; and iv) is sensitive to internal segment size variance.

To solve  $P_k$ ’s issues, Pevzner and Hearst (2002, pp. 10) proposed a modification referred to as WindowDiff (WD). Its major difference is in how it decides to penalized windows: within a window, if the number of boundaries in the manual segmentation ( $M_{ij}$ ) differs from the number of boundaries in the automatic segmentation ( $A_{ij}$ ), then a penalty is given. The ratio of penalties over windows then represents the degree of error between the segmentations, as in Equation 1. This change better allowed WD to: i) penalize FPs and FNs more equally;<sup>3</sup> ii) Not skip full misses; iii) Less harshly penalize near misses; and iv) Reduce its sensitivity to internal segment size variance.

$$WD(M, A) = \frac{1}{N - k} \sum_{i=1, j=i+k}^{N-k} (|M_{ij} - A_{ij}| > 0) \quad (1)$$

WD did not, however, solve all of the issues related to window-based segmentation comparison. WD, and inherently  $P_k$ : i) Penalize errors less at the beginning and end of segmentations (Lamprier et al., 2007); ii) Are biased towards favouring automatic segmentations with either few or tightly-clustered boundaries (Niekrasz and Moore, 2010); iii) Calculate window size  $k$  inconsistently;<sup>4</sup> iv) Are not symmetric<sup>5</sup> (meaning that they cannot be used to produce a pairwise mean of multiple manual segmentations<sup>6</sup>).

Segmentation Similarity (S; Fournier and Inkpen 2012, pp. 154–156) took a different approach to comparing segmentations. Instead of using windows, the work proposes a new restricted edit distance called *boundary edit distance* which differentiates between full and near misses. S then

<sup>2</sup>I.e., a boundary present in the manual but not the automatic segmentation, and the reverse for a false positive.

<sup>3</sup>Georgescul et al. (2006, p. 48) noted that WD interprets a near miss as a FP probabilistically more than as a FN.

<sup>4</sup> $k$  must be an integer, but half of a mean may be a fraction, thus rounding must be used, but no rounding method is specified. It is also not specified whether  $k$  should be set once during a study or recalculated for each comparison—this work assumes the latter.

<sup>5</sup>Window size is calculated only upon the manual segmentation, meaning that one must be a manual and other an automatic segmentation.

<sup>6</sup>This also means that WD and  $P_k$  cannot be adapted to compute inter-coder agreement coefficients.

normalizes the counts of full and near misses identified by boundary edit distance, as shown in Equation 2, where  $s_a$  and  $s_b$  are the segmentations,  $n_t$  is the maximum distance that boundaries may span to be considered a near miss,  $\text{edits}(s_a, s_b, n_t)$  is the edit distance, and  $\text{pb}(D)$  is the number of potential boundaries in a document  $D$  ( $\text{pb}(D) = |D| - 1$ ).

$$S(s_a, s_b, n_t) = 1 - \frac{|\text{edits}(s_a, s_b, n_t)|}{\text{pb}(D)} \quad (2)$$

Boundary edit distance models full misses as the addition/deletion of a boundary, and near misses as  $n$ -wise transpositions. An  $n$ -wise transposition is the act of swapping the position of a boundary with an empty position such that it matches a boundary in the segmentation compared against (up to a spanning distance of  $n_t$ ).  $S$  also scales the severity of a near miss by the distance over which it is transposed, allowing it to scale the penalty of a near misses much like WD.  $S$  is also symmetric, allowing it to be used in pairwise means and inter-coder agreement coefficients.

The usage of an edit distance that supported transpositions to compare segmentations was an advancement over window-based methods, but boundary edit distance and its normalization  $S$  are not without problems, specifically: i) This edit distance uses string reversals ( $ABCD \implies DCBA$ ) to perform transpositions, making it cumbersome to analyse individual pairs of boundaries between segmentations; ii)  $S$  is sensitive to variations in the total size of a segmentation, leading it to favour very sparse segmentations with few boundaries; iii)  $S$  produces cosmetically high values, making it difficult to interpret and causing over-estimation of inter-coder agreement. In this work, these deficiencies are demonstrated and a new set of metrics are proposed as replacements.

## 2.2 Inter-Coder Agreement

Inter-coder agreement coefficients are used to measure whether a group of human judges (i.e. coders) agree with each other greater than chance. Such coefficients are used to determine the reliability and replicability of the coding scheme and instructions used to collect manual codings (Carletta, 1996). Although direct interpretation of such coefficients is difficult, they are an invaluable tool when comparing segmentation data that has been collected with differing labels and when estimating the replicability of a study. A variety of inter-

coder agreement coefficients exist, but this work focuses upon a selection of those discussed by Artstein and Poesio (2008), specifically: Scott's  $\pi$  (Scott, 1955) Fleiss' multi- $\pi$  ( $\pi^*$ , Fleiss 1971)<sup>7</sup>, Cohen's  $\kappa$  (Cohen, 1960), and multi- $\kappa$  ( $\kappa^*$ , Davies and Fleiss 1982). Their general forms are shown in Equation 3, where  $A_a$  represents actual agreement, and  $A_e$  expected (i.e., chance) agreement between coders.

$$\kappa, \pi, \kappa^*, \text{ and } \pi^* = \frac{A_a - A_e}{1 - A_e} \quad (3)$$

When calculating agreement between manual segmenters, boundaries are considered labels and their positions the decisions. Unfortunately, because of the frequency of near misses that occur in segmentation, using such labels and decisions causes inter-coder agreement coefficients to drastically underestimate actual agreement—much like how automatic segmenter performance is underestimated when segmentation is treated as a binary classification problem. Hearst (1997, pp. 53–54) attempted to adapt  $\pi^*$  to award partial credit for near misses by using the percentage agreement metric of Gale et al. (1992, p. 254) to compute actual agreement—which conflates multiple manual segmentations together according to whether a majority of coders agree upon a boundary or not. Unfortunately, such a method of computing agreement grossly inflates results, and “the statistic itself guarantees at least 50% agreement by only pairing off coders against the majority opinion” (Isard and Carletta, 1995, p. 63).

Fournier and Inkpen (2012, pp. 154–156) proposed using pairwise mean  $S$  for actual agreement to allow inter-coder agreement coefficients to award partial credit for near misses. Unfortunately, because  $S$  produces cosmetically high values, it also causes inter-coder agreement coefficients to drastically overestimates actual agreement. This work demonstrates this deficiency and proposes and evaluates a solution.

## 3 A New Proposal for Edit-Based Text Segmentation Evaluation

In this section, a new boundary edit distance based segmentation metric and confusion matrix is proposed to solve the deficiencies of  $S$  for both segmentation comparison and inter-coder agreement.

<sup>7</sup>Sometimes referred to as  $K$  (Siegel and Castellan, 1988).

### 3.1 Boundary Edit Distance

In this section, Boundary Edit Distance (BED; as proposed in Fournier and Inkpen 2012, pp. 154–156) is introduced in more detail, and a few terminological and conceptual changes are made.

Boundary Edit Distance uses three main edit operations to model segmentation differences:

- Additions/deletions (AD; referred to originally as substitutions) for full misses;
- Substitutions (S; not shown for brevity) for confusing one boundary type with another;
- $n$ -wise transpositions (T) for near misses.

These edit operations are symmetric and operate upon the set of boundaries that occur at each potential boundary position in a pair of segmentations. An example of how these edit operations are applied<sup>8</sup> is shown in Figure 1, where a near miss (T), a matching pair of boundaries (M), and two full misses (ADs) are shown with the maximum distance that a transposition can span ( $n_t$ ) set to 2 potential boundaries (i.e., only adjacent positions can be transposed).

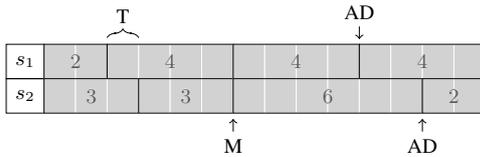


Figure 1: Boundary edit operations

In Figure 1, the location of the errors is clearly shown. Importantly, however, pairs of boundaries between the segmentations can be seen that represent the decisions made, and the correctness of these decisions. Imagine that  $s_1$  is a manual segmentation, and  $s_2$  is an automatic segmenter’s hypothesis. The transposition is a partially correct decision, or boundary pair. The match is a correct boundary pair. The additions/deletions, however, could be one of two erroneous decisions: to not place an expected boundary (FN), or to place a superfluous boundary (FP).<sup>9</sup>

This work proposes assigning a correctness score for each boundary pair/decision (shown in Table 1) and then using the mean of this score as a normalization of boundary edit distance. This interpretation intuitively relates boundary edit distance to coder judgements, making it ideal for

<sup>8</sup>A complete explanation of Boundary Edit Distance is detailed in Fournier (2013, Section 4.1.2).

<sup>9</sup>Also note that the ADs are close together, and if  $n_t > 2$ , then they would be considered a T, and not two ADs—this is one way to award partial credit for near misses.

calculating actual agreement in inter-coder agreement coefficients and comparing segmentations.

Pair	Correctness
Match	1
Addition/deletion	0
Transposition	$1 - w_{t\_span}(T_e, n_t)$
Substitution	$1 - w_{s\_ord}(S_e, T_b)$

Table 1: Correctness of boundary pair

### 3.2 Boundary Similarity

The new boundary edit distance normalization proposed herein is referred to as *boundary similarity* (B). Assuming that boundary edit distance produces sets of edit operations where  $A_e$  is the set of additions/deletions,  $T_e$  the set of  $n$ -wise transpositions,  $S_e$  the set of substitutions, and  $B_M$  the set of matching boundary pairs, boundary similarity can be defined as shown in Equation 4—one minus the incorrectness of each boundary pair over the total number of boundary pairs.

$$B(s_1, s_2, n_t) = 1 - \frac{|A_e| + w_{t\_span}(T_e, n_t) + w_{s\_ord}(S_e, T_b)}{|A_e| + |T_e| + |S_e| + |B_M|} \quad (4)$$

This form, one minus a penalty function, was chosen so that it was easier to compare against other penalty functions considered (not shown here for brevity). This normalization was also chosen because it is equivalent to mean boundary pair correctness and so that it ranges in value from 0 to 1. In the worst case, a segmentation comparison will result in no matches, no near misses, no substitutions, and  $X$  full misses, i.e.,  $|A_e| = X$  and all other terms in Equation 4 are zero, meaning that:

$$\begin{aligned} B &= 1 - \frac{X + 0 + 0}{X + 0 + 0 + 0} \\ &= 1 - X/X = 1 - 1 = 0 \end{aligned}$$

In the best case, a segmentation comparison will result in  $X$  matches, no near misses, no substitutions, and no full misses, i.e.,  $|B_M| = X$  and all other terms in Equation 4 are zero, meaning that:

$$\begin{aligned} B &= 1 - \frac{0 + 0 + 0}{0 + 0 + 0 + X} \\ &= 1 - 0/X = 1 - 0 = 1 \end{aligned}$$

For all other scenarios, varying numbers of matches, near misses, substitutions and full misses will result in values of B between 0 and 1.

Equation 4 takes two segmentations (in any order), and the maximum transposition spanning distance ( $n_t$ ). This distance represents the greatest offset between boundary positions that could be considered a near miss and can be used to scale

the severity of a near miss. A variety of scaling functions could be used, and this work arbitrarily chooses a simple fraction to represent each transposition’s severity in terms of its distance from its paired boundary over  $n_t$  plus a constant  $w_t$  (0 by default), as shown in Equation 5.

$$w_t\text{-span}(T_e, n_t) = \sum_{j=1}^{|T_e|} \left( w_t + \frac{\text{abs}(T_e[j][1] - T_e[j][2])}{n_t - 1} \right) \quad (5)$$

If multiple boundary types are used, then substitution edit operations would occur when one boundary type was confused with another. Assigning each boundary type  $t_b \in T_b$  a number on an ordinal scale, substitutions can be weighted by their distance on this scale over the maximum distance plus a constant  $w_s$  (0 by default), as shown in Equation 6.

$$w_s\text{-ord}(S_e, T_b) = \sum_{j=1}^{|S_e|} \left( w_s + \frac{\text{abs}(S_e[j][1] - S_e[j][2])}{\max(T_b) - \min(T_b)} \right) \quad (6)$$

These scaling functions allow for edit penalties to range from 0 to  $w_s/t$  plus some linear distance.

### 3.3 A Confusion Matrix for Segmentation

The mean correctness of each pair (i.e., B) gives an indication of just how similar one segmentation is to another, but what if one wants to identify some specific attributes of the performance of an automatic segmenter? Is the segmenter confusing one boundary type with another, or is it very precise but has poor recall? The answers to these questions can be obtained by looking at text segmentation as a multi-class classification problem.

This work proposes using a task’s set of boundary types ( $T_b$ ) and the lack of a boundary ( $\emptyset$ ) to represent the set of segmentation classes in a boundary classification problem. Using these classes, a confusion matrix (defined in Equation 7) can be created which sums boundary pair correctness so that information-retrieval metrics can be calculated that award partial credit to near misses by scaling edits operations.

$$\text{CM}(a, p) = \begin{cases} |B_{M,a}| + w_s\text{-ord}(S_e^{a,p}, T_b) \\ \quad + w_t\text{-span}(T_e^{a,p}, n_t) & \text{if } a = p \\ w_s\text{-ord}(S_e^{a,p}, T_b) \\ \quad + w_t\text{-span}(T_e^{a,p}, n_t) & \text{if } a \neq p \\ |A_{e,a}| & \text{if } p = \emptyset \\ |A_{e,p}| & \text{if } a = \emptyset \end{cases} \quad (7)$$

An example confusion matrix is shown in Figure 2 from which IR metrics such as precision, recall, and  $F_\beta$ -measure can be computed (referred to as B-precision, B-recall, etc.).

		Actual	
		B	Non-B
Predicted	B	CM(1, 1)	CM( $\emptyset$ , 1)
	Non-B	CM(1, $\emptyset$ )	CM( $\emptyset$ , $\emptyset$ )

Figure 2: Example confusion matrix ( $T_b = \{1\}$ )

### 3.4 B-Based Inter-coder Agreement

Fournier and Inkpen (2012, p. 156–157) adapted four inter-coder agreement formulations provided by Artstein and Poesio (2008) to use S to award partial credit for near misses, but because S produces cosmetically high agreement values they grossly overestimate agreement. To solve this issue, this work instead proposes using micro-average B (i.e., mean boundary pair correctness over all documents and codings compared) to solve this issue (demonstrated in §5) because it does not over-estimate actual agreement (demonstrated in §4 and 5).

## 4 Discussion of Segmentation Metrics

Before analysing how each metric compares to each other upon a large data set, it would be useful to investigate how they act on a smaller scale. To that end, this section discusses how each metric interprets a set of hypothetical segmentations of an excerpt of a poem by Coleridge (1816, pp. 55–58) titled *Kubla Khan* (shown in Figure 3)—chosen arbitrarily for its brevity (and beauty). These segmentations are topical and at the line-level.

1. In Xanadu did Kubla Khan
2. A stately pleasure-dome decree:
3. Where Alph, the sacred river, ran
4. Through caverns measureless to man
5. Down to a sunless sea.
6. So twice five miles of fertile ground
7. With walls and towers were girdled round:
8. And here were gardens bright with sinuous rills,
9. Where blossomed many an incense-bearing tree;
10. And here were forests ancient as the hills,
11. Enfolding sunny spots of greenery.

Figure 3: Excerpt from the poem *Kubla Khan* (Coleridge, 1816, pp. 55–58) with line numbers

Topical segmentations of this poem are difficult to produce because there is still some structural form (i.e., punctuation) which may dictate where a boundary lies, but the imagery, places, times, and subjects of the poem appear to twist and wind like a vision in a dream. Thus, placing a topical boundary in this text is a highly subjective task. One hypothetical topical segmentation of the excerpt is shown in Figure 4. In this section, a variety of

contrived automatic segmentations are compared to this manual segmentation to illustrate how each metric reacts to different mistakes.

Lines	Description
1–2	Kubla Khan and his decree
3–5	Waterways
6–11	Fertile ground and greenery

Figure 4: A hypothetical manual segmentation

Assuming that Figure 4 represents an acceptable manual segmentation ( $m$ ), how would each metric react to an automatic segmentation ( $a$ ) that combines the segments 1–2 and 3–5 together? This would represent a full miss, or a false negative, as shown in Figure 5. S interprets these segmentations as being quite similar, yet, the automatic segmentation is missing a boundary. B and  $1-WD$ ,<sup>10</sup> in this case, better reflect this error.

$m$	■	■	■	■	■	■	■	■	■	■	$s$	$B$	$1-WD$
$a$	■	■	■	■	■	■	■	■	■	■	0.9	0.5	0.777
$k = 2$													

Figure 5: False negative

How would each metric react to an automatic segmentation that is very close to placing the boundaries correctly, but makes the slight mistake of thinking that the segment on waterways (3–5) ends a bit too early? This would represent a near miss, as shown in Figure 6. S and  $1-WD$  incorrectly interpret this error as being equivalent to the previous false negative—a troubling result. Segmentation comparison metrics should be able to discern between the full and a near miss shown in these two figures, and an automatic segmenter that nearly misses a boundary should be awarded a better score than one which fully misses a boundary—B recognizes this and awards the near miss a higher score.

$m$	■	■	■	■	■	■	■	■	■	■	$s$	$B$	$1-WD$
$a$	■	■	■	■	■	■	■	■	■	■	0.9	0.75	0.777
$k = 2$													

Figure 6: Near miss

How would each metric react to an automatic segmentation that adds an additional boundary between line 8 and 9? This would not be ideal because such a boundary falls in the middle of a cohesive description of a garden, representing

<sup>10</sup>WD is reported as  $1-WD$  because WD is normally a penalty metric where a value of 0 is ideal, unlike S and B. Additionally,  $k = 2$  for all examples in this section because WD computes  $k$  from the manual segmentation  $m$ , which does not change in these examples.

a full miss, or false positive, as in Figure 7. S and  $1-WD$  incorrectly interpret this error as being equivalent to the previous two errors—an even more troubling result. In this case, there are two matching boundaries and a pair that do not match, which is arguably preferable to the full miss and one match in Figure 5, but not to the match and near miss in Figure 6. B recognizes this, and awards a higher score to this automatic segmenter than that in Figure 5, but below Figure 6.

$m$	■	■	■	■	■	■	■	■	■	■	$s$	$B$	$1-WD$
$a$	■	■	■	■	■	■	■	■	■	■	0.9	0.666	0.777
$k = 2$													

Figure 7: False positive

How would each metric react to an automatic segmentation that compensates for its lack of precision by spuriously adding boundaries in clusters around where it thinks that segments should begin or end? This is shown in Figure 8. This kind of behaviour is finally penalized differently by S and  $1-WD$  (unlike the other errors shown in this section), but it only barely results in a dip in their values. B also penalizes this behaviour, but does so much more harshly—in B’s interpretation, this is as egregious as committing a false negative (e.g., Figure 5)—an arguably correct interpretation, if the evaluation desires to maximize similarity with a manual segmentation.

$m$	■	■	■	■	■	■	■	■	■	■	$s$	$B$	$1-WD$
$a$	■	■	■	■	■	■	■	■	■	■	0.8	0.5	0.666
$k = 2$													

Figure 8: Cluster of false positives

These short demonstrations of how S, B, and  $1-WD$  interpret error should lead one to conclude that: i) WD can penalize near misses to the same degree as full misses—overly harshly; ii) Both S and WD are not very discriminating when small segments are analysed; and iii) B is the only one of the three metrics that is able to often discriminate between these situations. B, if used to rank these automatic segmenters, would rank them from best to worst performing as: the near miss, false positive, and then a tie between the false negative and cluster of false positives—a reasonable ranking in the context of an evaluation seeking similarity with a manual segmentation.

## 5 Segmentation Agreement

Having a bit more confidence in B compared to S and WD on a small scale from the previous sec-

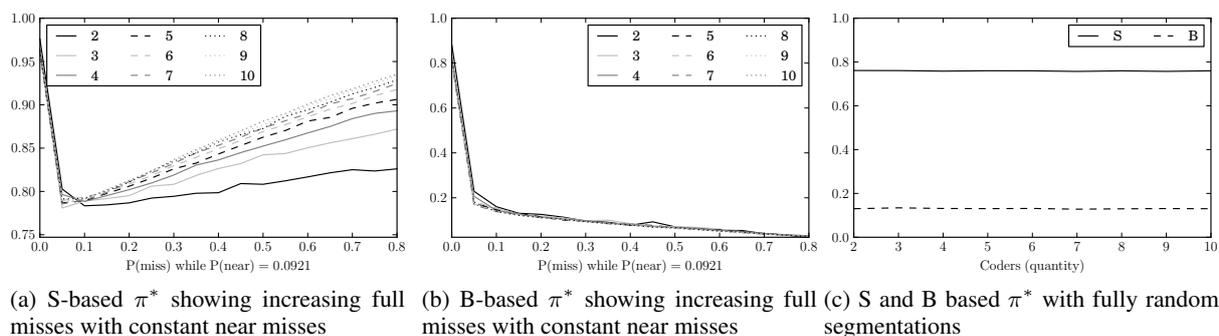


Figure 9: Artificial data sets illustrating how  $\pi$  adapted to use either S or B reacts to increasing full misses and random segmentations and varying numbers of coders

tion, it makes sense to analyse some larger data sets. Two such data sets are The Stargazer data set collected by Hearst (1997) and The Moonstone data set collected by Kazantseva and Szpakowicz (2012). Both are linear topical segmentations at the paragraph level with only one boundary type, but that is where their similarities end.

The Stargazer text is a science magazine article titled “Stargazers look for life” (Baker, 1990) segmented by 7 coders and was one of twelve articles chosen for its length (between 1,800 and 2,500 words) and for having little structural demarcation. “The Moonstone” is a 19<sup>th</sup> century romance novel by Collins (1868) segmented by 4–6 coders per chapter; of its 23 chapters, 2 were coded in a pilot study and another 20 were coded individually by 27 undergraduate English students in 5 groups.

For the Stargazer data set, using S-based  $\pi^*$ , an inter-coder agreement coefficient of 0.7562 is obtained—a reasonable level by content analysis standards. Unfortunately, this value is highly inflated, and B-based  $\pi^*$  gives a much more conservative coefficient at 0.4405. For the Moonstone data set, the agreement coefficients for each group of 4–6 coders using S-based  $\pi^*$  is again over-inflated at 0.91, 0.92, 0.90, 0.94, 0.83. B-based  $\pi^*$  instead reports that the coefficients should be 0.20, 0.18, 0.40, 0.38, 0.23.

Which of these coefficients should be trusted? Is agreement in these data sets high or low? To help answer that, this work looks at how the coders in the data sets behaved. If the segmenters in the Moonstone data set truly agreed with each other, then they should have all behaved similarly. One measure of coder behaviour is the frequency that they placed boundaries (normalized by their opportunity to place boundaries, i.e. the sum of the potential boundaries in the chapters that each segmented). This normalized frequency is shown per

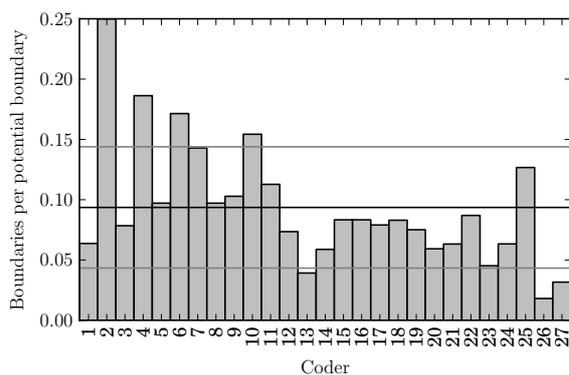


Figure 11: Normalized boundaries placed by each coder in the Moonstone data set (with mean $\pm$ SD)

coder in Figure 11 for The Moonstone data set, along with bars indicating the mean and one standard deviation above and below. As can be seen, the coders fluctuated wildly in the frequency with which they placed boundaries—some (e.g., coder 7) to degrees exceeding 2 standard deviations. The Moonstone data set as a whole does not exhibit coders who behaved similarly, supporting the assertion by B-based  $\pi^*$  that these coders do not agree well (though pockets of agreement exist).

How can it be demonstrated that S-based agreement over-estimates agreement, and B-based agreement does not? One way to demonstrate this is through simulation. By estimating parameters from the large Moonstone data set such as the distribution of internal segment sizes produced by all coders, a random segmentation of the novel with similar characteristics can be created. From this single random segmentation, other segmentation can be created with a probability of either placing an offset boundary (i.e., a near miss) or placing an extra/omitting a boundary (i.e., a full miss)—a pseudo-coding. Manipulating these probabilities and keeping the probability of a near miss at a constant natural level should produce a slowly declin-

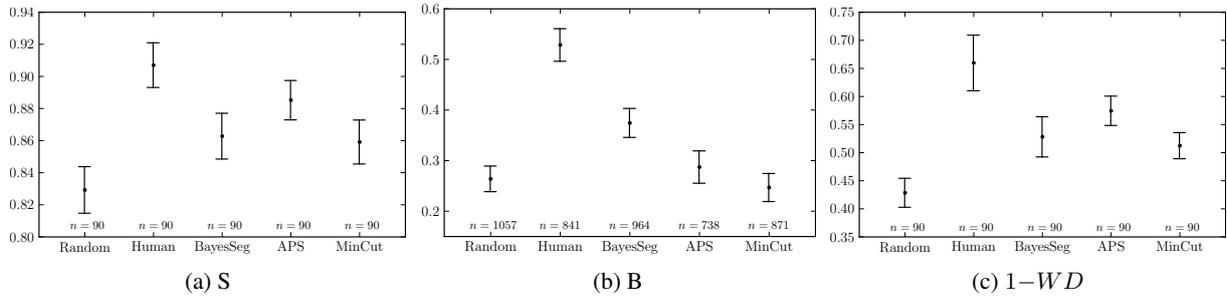


Figure 10: Mean performance of 5 segmenters using varying metrics with 95% confidence intervals

ing amount of agreement which is unaffected by the number of pseudo-coders. This is not apparent, however, for S-based  $\pi^*$  in Figure 9a; as the probability of a full miss increases, agreement appears to rise and varies depending upon the number of pseudo-coders. B-based  $\pi^*$  however shows declining agreement and little to no variation depending upon the number of pseudo-coders, as shown in Figure 9b.

If instead of creating pseudo-coders from a random segmentation a series of random segmentations with the same parameters were generated, a properly functioning inter-coder agreement coefficient should report some agreement (due to the similar parameters used to create the segmentations) but it should be quite low. Figure 9c shows this, and that S-based  $\pi^*$  drastically over-estimates what should be very low agreement whereas B-based  $\pi^*$  properly reports low agreement.

From these demonstrations, it is evident that S-based inter-coder agreement coefficients drastically over-estimate agreement, as does S itself in pairwise mean form. B-based coefficients, however, properly discriminate between levels of agreement regardless of the number of coders and do not over-estimate.

## 6 Evaluation of Automatic Segmenters

Having looked at how S, WD, and B perform at a small scale in §4 and on larger data set in §5, this section demonstrates the use of these metrics to evaluate some automatic segmenters. Three automatic segmenters were trained—or had their parameters estimated upon—The Moonstone data set, including MinCut; (Malioutov and Barzilay, 2006), BayesSeg; (Eisenstein and Barzilay, 2008), and APS (Kazantseva and Szpakowicz, 2011).

To put this evaluation into context, an upper and lower bound were also created comprised of a random coder from the manual data (Human) and a

random segmenter (Random), respectively. These automatic segmenters, and the upper and lower bounds, were created, trained, and run by another researcher (Anna Kazantseva) with their labels removed during the development of the metrics detailed herein (to improve the impartiality of these analyses). An ideal segmentation evaluation metric should, in theory, place the three automatic segmenters between the upper and lower bounds in terms of performance if the metrics, and the segmenters, function properly.

The mean performance of the upper and lower bounds upon the test set of the Moonstone data set using S, B, and WD are shown in Figure 10a–10c along with 95% confidence intervals. Despite the difference in the scale of their values, both S and WD performed almost identically, placing the three automatic segmenters between the upper and lower bounds as expected. For S, statistically significant differences<sup>11</sup> ( $\alpha = 0.05$ ) were found between all segmenters except between APS–human and MinCut–BayesSeg, and WD could only find significant differences between the automatic segmenters and the upper and lower bounds. B, however, shows a marked deviation, and places MinCut and APS statistically significantly below the random baseline with only BayesSeg between the upper and lower bounds—to a significant degree.

Why would pairwise mean B act in such an unexpected manner? The answer lies in a further analysis using the confusion matrix proposed earlier to calculate B-precision and B-recall (as shown in Table 2). From the values in Table 2, all three automatic segmenters appear to have B-precision above the baseline and below the upper bound, but the B-recall of both APS and MinCut is below that of the random baseline (illustrated

<sup>11</sup>Using Kruskal-Wallis rank sum multiple comparison tests (Siegel and Castellan, 1988, pp. 213-214) for S and WD and the Wilcoxon-Nemenyi-McDonald-Thompson test (Hollander and Wolfe, 1999, p. 295) for B.

	B	$n$	B-P	B-R	B-F <sub>1</sub>	TP	FP	FN	TN
Random	$0.2640 \pm 0.0129$	1057	0.3991	0.4673	0.4306	279.0	420	318	4236.0
Human	$0.5285 \pm 0.0164$	841	0.6854	0.7439	0.7135	444.5	204	153	4451.5
BayesSeg	$0.3745 \pm 0.0146$	964	0.5247	0.6224	0.5694	361.0	327	219	4346.0
APS	$0.2873 \pm 0.0163$	738	0.6773	0.3403	0.4530	212.0	101	411	4529.0
MinCut	$0.2468 \pm 0.0141$	871	0.4788	0.3496	0.4041	215.0	234	400	4404.0

Table 2: Mean performance of 5 segmenters using micro-average B, B-precision (B-P), B-recall (B-R), and B-F<sub>β</sub>-measure (B-F<sub>1</sub>) along with the associated confusion matrix values for 5 segmenters

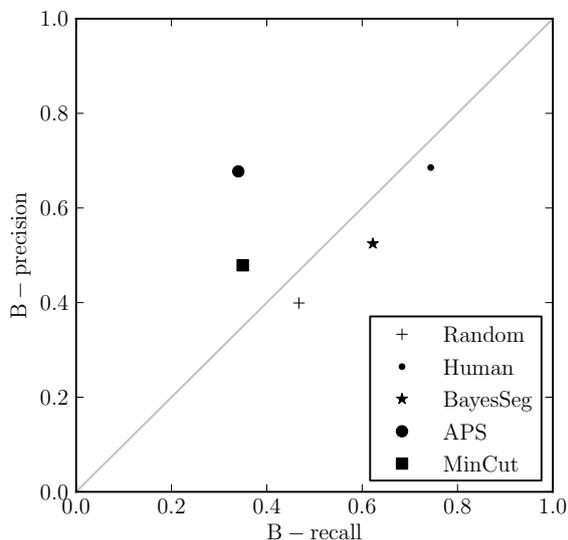


Figure 12: Mean B-precision versus B-recall of 5 automatic segmenters

in Figure 12). These automatic segmenters were developed and performance tuned using WD, thus it would be expected that they would perform as they did according to WD, but the evaluation using B highlights WD’s bias towards sparse segmentations (i.e., those with low B-recall)—a failing that S also appears to share. Mean B shows an unbiased ranking of these automatic segmenters in terms of the upper and lower bounds. B, then, should be preferred over S and WD for an unbiased segmentation evaluation that assumes that similarity to a human solution is the best measure of performance for a task.

## 7 Conclusions

In this work, a new segmentation evaluation metric, referred to as *boundary similarity* (B) is proposed as an unbiased metric, along with a boundary-edit-distance-based (BED-based) confusion matrix to compute predictably biased IR metrics such as precision and recall. Additionally, a method of adapting inter-coder agreement coefficients to award partial credit for near misses is proposed that uses B as opposed to S for actual agreement so as to not over-estimate agreement.

B overcomes the cosmetically high values of S and, the bias towards segmentations with few or tightly-clustered boundaries of WD—manifesting in this work as a bias towards precision over recall for both WD and S. When such precision is desirable, however, B-precision can be computed from a BED-based confusion matrix, along with other IR metrics. WD and  $P_k$  should not be preferred because their biases do not occur consistently in all scenarios, whereas BED-based IR metrics offer expected biases built upon a consistent, edit-based, interpretation of segmentation error.

B also allows for an intuitive comparison of boundary pairs between segmentations, as opposed to the window counts of WD or the simplistic edit count normalization of S. When an unbiased segmentation evaluation metric is desired, this work recommends the usage of B and the use of an upper and lower bound to provide context. Otherwise, if the evaluation of a segmentation task requires some biased measure, the predictable bias of IR metrics computed from a BED-based confusion matrix is recommended. For all evaluations, however, a justification for the biased/unbiased metrics used should be given, and more than one metric should be reported so as to allow a reader to ascertain for themselves whether a particular automatic segmenter’s bias in some manner is cause for concern or not.

## 8 Future Work

Future work includes adapting this work to analyse hierarchical segmentations and using it to attempt to explain the low inter-coder agreement coefficients reported in topical segmentation tasks.

## Acknowledgements

I would like to thank Anna Kazantseva for her invaluable feedback and data. Additionally, I would like to thank my thesis committee members—Stan Szpakowicz, James Green, and Xiaodan Zhu—for their feedback along with my supervisor Diana Inkpen and colleague Martin Scaiano.

## References

- Artstein, Ron and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics* 34(4):555–596.
- Baker, David. 1990. Stargazers look for life. *South Magazine* 117:76–77.
- Beeferman, Doug and Adam Berger. 1999. Statistical models for text segmentation. *Machine Learning* 34:177–210.
- Carletta, Jean. 1996. Assessing Agreement on Classification Tasks: The Kappa Statistic. *Computational Linguistics* 22(2):249–254.
- Chang, Pi-Chuan, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the Third Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 224–232.
- Cohen, Jacob. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement* 20:37–46.
- Coleridge, Samuel Taylor. 1816. *Christabel, Kubla Khan, and the Pains of Sleep*. John Murray.
- Collins, Wilkie. 1868. *The Moonstone*. Tinsley Brothers.
- Davies, Mark and Joseph L. Fleiss. 1982. Measuring agreement for multinomial data. *Biometrics* 38:1047–1051.
- Eisenstein, Jacob. 2009. Hierarchical text segmentation from multi-scale lexical cohesion. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 353–361.
- Eisenstein, Jacob and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Morristown, NJ, USA, pages 334–343.
- Fleiss, Joseph L. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin* 76:378–382.
- Fournier, Chris and Diana Inkpen. 2012. Segmentation Similarity and Agreement. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 152–161.
- Fournier, Christopher. 2013. *Evaluating Text Segmentation*. Master's thesis, University of Ottawa.
- Franz, Martin, J. Scott McCarley, and Jian-Ming Xu. 2007. User-oriented text segmentation evaluation measure. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, Stroudsburg, PA, USA, pages 701–702.
- Gale, William, Kenneth Ward Church, and David Yarowsky. 1992. Estimating upper and lower bounds on the performance of word-sense disambiguation programs. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 249–256.
- Georgescul, Maria, Alexander Clark, and Susan Armstrong. 2006. An analysis of quantitative aspects in the evaluation of thematic segmentation algorithms. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 144–151.
- Haghighi, Aria and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL '09, pages 362–370.
- Hearst, Marti A. 1993. TextTiling: A Quantitative Approach to Discourse. Technical report, University of California at Berkeley, Berkeley, CA, USA.
- Hearst, Marti A. 1997. TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Computational Linguistics* 23:33–64.
- Hollander, Myles and Douglas A. Wolfe. 1999.

- Nonparametric Statistical Methods*. John Wiley & Sons, 2nd edition.
- Isard, Amy and Jean Carletta. 1995. Replicability of transaction and action coding in the map task corpus. In *AAAI Spring Symposium: Empirical Methods in Discourse Interpretation and Generation*. pages 60–66.
- Kazantseva, Anna and Stan Szpakowicz. 2011. Linear Text Segmentation Using Affinity Propagation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland, UK., pages 284–293.
- Kazantseva, Anna and Stan Szpakowicz. 2012. Topical Segmentation: a Study of Human Performance. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 211–220.
- Lamprier, Sylvain, Tassadit Amghar, Bernard Levrat, and Frederic Saubion. 2007. On evaluation methodologies for text segmentation algorithms. In *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence*. IEEE Computer Society, Washington, DC, USA, volume 2, pages 19–26.
- Litman, Diane J. and Rebecca J. Passonneau. 1995. Combining multiple knowledge sources for discourse segmentation. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 108–115.
- Malioutov, Igor and Regina Barzilay. 2006. Minimum cut model for spoken lecture segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 25–32.
- Niekrasz, John and Johanna D. Moore. 2010. Unbiased discourse segmentation evaluation. In *Proceedings of the IEEE Spoken Language Technology Workshop, SLT 2010*. IEEE 2010, pages 43–48.
- Oh, Hyo-Jung, Sung Hyon Myaeng, and Myung-Gil Jang. 2007. Semantic passage segmentation based on sentence topics for question answering. *Information Sciences* 177(18):3696–3717.
- Passonneau, Rebecca J. and Diane J. Litman. 1993. Intention-based segmentation: human reliability and correlation with linguistic cues. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 148–155.
- Pevzner, Lev and Marti A. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics* 28:19–36.
- Reynar, Jeffrey C. and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the 5th Conference on Applied Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 16–19.
- Scott, William A. 1955. Reliability of content analysis: The case of nominal scale coding. *Public Opinion Quarterly* 19:321–325.
- Siegel, Sidney and N. J. Castellan. 1988. *Nonparametric Statistics for the Behavioral Sciences*, McGraw-Hill, New York, USA, chapter 9.8. 2nd edition.
- Sirts, Kairit and Tanel Alumäe. 2012. A Hierarchical Dirichlet Process Model for Joint Part-of-Speech and Morphology Induction. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 407–416.
- Stoyanov, Veselin and Claire Cardie. 2008. Topic identification for fine-grained opinion analysis. In *Proceedings of the 22nd International Conference on Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, pages 817–824.

# Crowd Prefers the Middle Path: A New IAA Metric for Crowdsourcing Reveals Turker Biases in Query Segmentation

**Rohan Ramanath\***

R. V. College of Engineering  
Bangalore, India  
ronramanath@gmail.com

**Monojit Choudhury**

Microsoft Research Lab India  
Bangalore, India  
monojitc@microsoft.com

**Kalika Bali**

Microsoft Research Lab India  
Bangalore, India  
kalikab@microsoft.com

**Rishiraj Saha Roy<sup>†</sup>**

Indian Institute of Technology Kharagpur  
Kharagpur, India  
rishiraj@cse.iitkgp.ernet.in

## Abstract

Query segmentation, like text chunking, is the first step towards query understanding. In this study, we explore the effectiveness of crowdsourcing for this task. Through carefully designed control experiments and Inter Annotator Agreement metrics for analysis of experimental data, we show that crowdsourcing may not be a suitable approach for query segmentation because the crowd seems to have a very strong bias towards dividing the query into roughly equal (often only two) parts. Similarly, in the case of hierarchical or nested segmentation, turkers have a strong preference towards balanced binary trees.

## 1 Introduction

*Text chunking* of Natural Language (NL) sentences is a well studied problem that is an essential pre-processing step for many NLP applications (Abney, 1991; Abney, 1995). In the context of Web search queries, *query segmentation* is similarly the first step towards analysis and understanding of queries (Hagen et al., 2011). The task in both the cases is to divide the sentence or the query into contiguous *segments* or chunks of words such that the words from a segment are related to each other more strongly than words from different segments (Bendersky et al., 2009). It is typically assumed that the segments are structurally and semantically coherent and, therefore, the information contained in them can be processed holistically.

\*The work was done during author’s internship at Microsoft Research Lab India.

<sup>†</sup>This author was supported by Microsoft Corporation and Microsoft Research India under the Microsoft Research India PhD Fellowship Award.

$f$	Pipe representation	Boundary var.
4	apply   first aid course   on line	1 0 0 1 0
3	apply first aid course   on line	0 0 0 1 0
2	apply first aid   course on line	0 0 1 0 0
1	apply   first aid   course   on line	1 0 1 1 0

Table 1: Example of flat segmentation by Turkers.  $f$  is the frequency of annotations; segment boundaries are represented by |.

$f$	Bracket representation	Boundary var.
4	((apply first) ((aid course) (on line)))	0 2 0 1 0
2	((apply (first aid)) course) (on line))	1 0 2 3 0
2	((apply ((first aid) course)) (on line))	2 0 1 3 0
1	(apply (((first aid) course) (on line)))	3 0 1 2 0
1	((apply (first aid)) (course (on line)))	1 0 2 1 0

Table 2: Example of nested segmentation by Turkers.  $f$  is the frequency of annotations.

A majority of work on query segmentation relies on manually segmented queries by human experts for training and evaluation of segmentation algorithms. These are typically small datasets and even with detailed annotation guidelines and/or close supervision, low Inter Annotator Agreement (IAA) remains an issue. For instance, Table 1 illustrates the variation in flat segmentation by 10 annotators. This confusion is mainly because the definition of a segment in a query is ambiguous and of an unspecified granularity. This is further compounded by the fact that other than easily recognizable and agreed upon segments such as Named Entities or Multi-Word Expressions, there is no established notion of linguistic grouping such as phrases and clauses in a query.

Although there is little work on the use of crowdsourcing for query segmentation (Hagen et al., 2011; Hagen et al., 2012), the idea that the

crowd could be a potential (and cheaper) source for reliable segmentation seems a reasonable assumption. The need for larger datasets makes this an attractive proposition. Also, a larger number of annotations could be appropriately distilled to obtain better quality segmentations.

In this paper we explore crowdsourcing as an option for query segmentation through experiments designed using Amazon Mechanical Turk (AMT)<sup>1</sup>. We compare the results against gold datasets created by trained annotators. We address the issues pertaining to disagreements due to both ambiguity and granularity and attempt to objectively quantify their role in IAA. To this end, we also conduct similar annotation experiments for NL sentences and randomly generated queries. While queries are not as structured as NL sentences they are not simply a set of random words. Thus, it is necessary to compare query segmentation to the über-structure of NL sentences as well as the unter-structure of random  $n$ -grams. This has important implications for understanding any inherent biases annotators may have as a result of the apparent lack of structure of the queries.

To quantify the effect of granularity on segmentation, we also ask annotators to provide hierarchical or nested segmentations for real and random queries, as well as sentences. Following Abney’s (1992) proposal for hierarchical chunking of NL, we ask the annotators to group *exactly two* words or segments at a time to recursively form bigger segments. The concept is illustrated in Fig. 1. Table 2 shows annotations from 10 Turkers. It is important to constrain the joining of exactly two segments or words at a time to avoid the issue of fuzziness in granularity. We shall refer to this style of annotation as *Nested segmentation*, whereas the non-hierarchical non-constrained chunking will be referred to as *Flat segmentation*.

Through statistical analysis of the experimental data we show that crowdsourcing may not be the best practice for query segmentation, not only because of ambiguity and granularity issues, but because there exist very strong biases amongst annotators to divide a query into two roughly equal parts that result in misleadingly high agreements. As a part of our analysis framework, we introduce a new IAA metric for comparison across flat and nested segmentations. This versatile metric can be

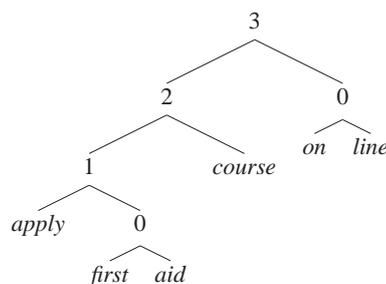


Figure 1: Nested Segmentation: Illustration.

readily adapted for measuring IAA for other linguistic annotation tasks, especially when done using crowdsourcing.

The rest of the paper is organized as follows. Sec 2 provides a brief overview of related work. Sec 3 describes the experiment design and procedure. In Sec 4, we introduce a new metric for IAA, that could be uniformly applied across flat and nested segmentations. Results of the annotation experiments are reported in Sec 5. In Sec 6, we analyze the possible statistical and linguistic biases in annotation. Sec 7 concludes the paper by summarizing the work and discussing future research directions. All the annotated datasets used in this research are freely available for non-commercial research purposes<sup>2</sup>.

## 2 Related Work

Query segmentation was introduced by Risvik et. al. (2003) as a possible means to improve Information Retrieval. Since then there has been a significant amount of research exploring various algorithms for this task and its use in IR (see Hagen et. al. (2011) for a survey). Most of the research and evaluation considers query segmentation as a process analogous to identification of phrases within a query which when put within double-quotes (implying exact matching of the quoted phrase in the document) leads to better IR performance. However, this is a very restricted view of the process and does not take into account the full potential of query segmentation.

A more generic notion of segments leads to diverse and ambiguous definitions, making its evaluation a hard problem (see Saha Roy et. al. (2012) for a discussion on issues with evaluation). Most automatic segmentation techniques (Bergsma and Wang, 2007; Tan and Peng, 2008; Zhang et al.,

<sup>2</sup>Related datasets and supplementary material can be accessed from <http://bit.ly/161Gkk9> or can be obtained by directly emailing the authors.

<sup>1</sup><http://www.mturk.com/mturk/welcome>

2009; Brenes et al., 2010; Hagen et al., 2011; Li et al., 2011) have so far been evaluated only against a small set of human-annotated queries (Bergsma and Wang, 2007). The reported low IAA for such datasets casts serious doubts on the reliability of annotation and the performance of the algorithms evaluated on them (Hagen et al., 2011; Saha Roy et al., 2012).

To address the problem of data scarcity, Hagen et al. (2011) have created larger annotated datasets through crowdsourcing<sup>3</sup>. However, in their approach the crowd is provided with a few (four) possible segmentations of a query to choose from (known through a personal communication with a authors). Thus, it presupposes an automatic process that can generate the correct segmentation of a query within top few options. It is far from obvious how to generate these initial segmentations in a reliable manner. This may also result in an over-optimistic IAA. An ideal segmentation should be based on the annotators' own interpretation of the query. Nevertheless, if large scale data has to be procured, crowdsourcing seems to be the only efficient and effective model for this task, and has been proven to be so for other IR and linguistic annotations; see Carvalho et al. (2011) for examples of crowdsourcing for IR resources and (Snow et al., 2008; Callison-Burch, 2009) for language resources.

In the context of NL text, segmentation has been traditionally referred to as *chunking* and is a well-studied problem. Abney (1991; 1992; 1995) defines a chunk as a sub-tree within a syntactic phrase structure tree corresponding to Noun, Prepositional, Adjectival, Adverbial and Verb Phrases. Similarly, Bharati et al (1995) defines it as Noun Group and Verb Group based only on local surface information. However, cognitive and annotation experiments for chunking of English (Abney, 1992) and other language text (Bali et al., 2009) have shown that native speakers agree on major clause and phrase boundaries, but may not do so on more fine-grained chunks. One important implication of this is that annotators are expected to agree more on the higher level boundaries for nested segmentation than the lower ones. We note that hierarchical query segmentation was proposed for the first time by Huang et al. (2010), where the authors recursively split a query (or its fragment) into exactly two parts and evaluate the

final output against human annotations.

### 3 Experiments

The annotation experiments have been designed to systematically study the various aspects of query segmentation. In order to verify the effectiveness and reliability of crowdsourcing, we designed an AMT experiment for flat segmentation of Web search queries. As a baseline, we would like to compare these annotations with those from human experts trained for the task. We shall refer to this baseline as the *Gold annotation set*. Since we believe that the issue of granularity could be the prime reason for previously reported low IAA for segmentation, we also designed AMT-based nested segmentation experiments for the same set of queries, and obtained the corresponding gold annotations.

Finally, to estimate the role of ambiguity inherent in the structure of Web search queries on IAA, we conducted two more control experiments, both through crowdsourcing. First, flat and nested segmentation of well-formed English, i.e., NL sentences of similar length distribution; and second, flat and nested segmentation of randomly generated queries. Higher IAA for NL sentences would lead us to conclude that ambiguity and lack of structure in queries is the main reason for low agreements. On the other hand high or comparable IAA for random queries would mean that annotations have strong biases.

Thus, we have the following four pairs of annotation experiments: flat and nested segmentation of queries from crowdsourcing, corresponding flat and nested gold annotations, flat and nested segmentation of English sentences from crowdsourcing, and flat and nested segmentations for randomly generated queries through crowdsourcing.

#### 3.1 Dataset

For our experiments, we need a set of Web search queries and well-formed English sentences. Furthermore, for generating the random queries, we will use search query logs to learn  $n$ -gram models. In particular, we use the following datasets:

**Q500, QG500:** Saha Roy et al. (2012) released a dataset of 500 queries, 5 to 8 words long, for evaluation of various segmentation algorithms. This dataset has flat segmentations from three annotators obtained under controlled experimental settings, and can be considered as *Gold* annota-

<sup>3</sup><http://www.webis.de/research/corpora>

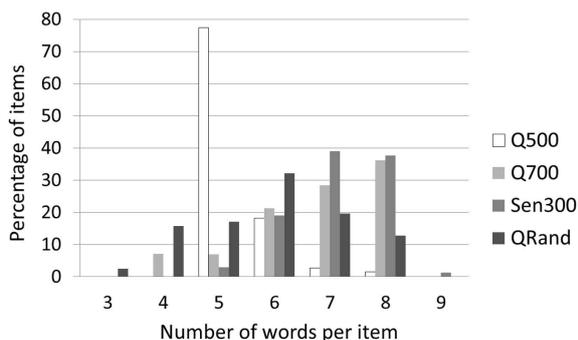


Figure 2: Length distribution of datasets.

tions. Hence, we select this set for our experiments as well. We procured the corresponding nested segmentation for these queries from two human experts, who are regular search engine users, between 20 and 30 years old, and familiar with various linguistic annotation tasks. They annotated the data under supervision. They were trained and paid for the task. We shall refer to the set of flat and nested gold annotations as **QG500**, whereas **Q500** will be reserved for AMT experiments.

**Q700:** Since 500 queries may not be enough for reliable conclusion and since the queries may not have been chosen specifically for the purpose of annotation experiments, we expanded the set with another 700 queries sampled from a slice of the query logs of Bing Australia<sup>4</sup> containing 16.7 million queries issued over a period of one month (May 2010). We picked, uniformly at random, queries that are 4 to 8 words long, have only English letters and numerals, and a high *click entropy* because “a query with a larger click entropy value is more likely to be an informational or ambiguous query” (Dou et al., 2008). **Q500** consists of tailish queries with frequency between 5 and 15 that have at least one multiword named entity; but unlike the case of **Q700**, click-entropy was not considered during sampling. As we shall see, this difference is clearly reflected in the results.

**S300:** We randomly selected 300 English sentences from a collection of full texts of public domain books<sup>5</sup> that were 5 to 15 words long, and checked them for well-formedness. This set will be referred to as **S300**.

**QRand:** Instead of generating search queries by throwing in words randomly, we thought it will be more interesting to explore annotation of

Parameter	Flat Details	Nested Details
Time needed: actual (allotted)	49 sec (10 min)	1 min 52 sec (15 min)
Reward per HIT	\$0.02	\$0.06
Instruction video duration	26 sec	1 min 40 sec
Turker qualification	Completion rate >100 tasks	
Turker approval rate	Acceptance rate >60 %	
Turker location	United States of America	

Table 3: Specifics of the HITs for AMT.

queries generated using  $n$ -gram models for  $n = 1, 2, 3$ . We estimated the models from the Bing Australia log of 16.7 million queries. We generated 250 queries each of desired length distribution using the 1, 2 and 3-gram models. We shall refer to these as **U250**, **B250**, **T250** (for Uni, Bi and Trigram) respectively, and the whole dataset as **QRand**. Fig. 2 shows the query and sentence length distribution for the various sets.

### 3.2 Crowdsourcing Experiments

We used AMT to get our annotations through crowdsourcing. Pilot experiments were carried out to test the instruction set and examples presented. Based on the feedback, the precise instructions for the final experiments were designed.

Two separate AMT Human Intelligence Tasks (HITs) were designed for flat and nested query segmentation. Also, the experiments for queries (**Q500+Q700**) were conducted separately from **S300** and **QRand**. Thus, we had six HITs in all. The concept of flat and nested segmentation was introduced to the Turkers with the help of examples presented in two short videos<sup>6</sup>. When in doubt regarding the meaning of a query, the Turkers were advised to issue the query on a search engine of their choice and find out its possible interpretation(s). Note that we intentionally kept definitions of flat and nested segmentation fuzzy because (a) it would require very long instruction manuals to cover all possible cases and (b) Turkers do not tend to read verbose and complex instructions. Table 3 summarizes other specifics of HITs.

*Honey pots* or trap questions whose answers are known a priori are often included in a HIT to identify turkers who are unable to solve the task appropriately leading to incorrect annotations. However, this trick cannot be employed in our case because there is no notion of an absolutely correct segmentation. We observe that even with unambiguous queries, even expert annotators may dis-

<sup>4</sup><http://www.bing.com/?cc=au>

<sup>5</sup><http://www.gutenberg.org>

<sup>6</sup>Flat: <http://youtu.be/eMeLjJivIh0>, Nested: <http://youtu.be/xE3rwANbFvU>

agree on some of the segment boundaries. Hence, we decided to include annotations from all the turkers, except for those that were syntactically ill-formed (e.g., non-binary nested segmentation).

## 4 Inter Annotator Agreement

*Inter Annotator Agreement* is the only way to judge the reliability of annotated data in absence of an end application. Therefore, before we can venture into analysis of the experimental data, we need to formalize the notion of IAA for flat and nested queries. The task is non-trivial for two reasons. First, traditional IAA measures are defined for a fixed set of annotators. However, for crowdsourcing based annotations, different annotators might have annotated different parts of the dataset. For instance, we observed that a total of 128 turkers have provided the flat annotations for **Q700**, when we had only asked for 10 annotations per query. Thus, on average, a turker has annotated only 7.81% of the 700 queries. In fact, we found that 31 turkers had annotated less than 5 queries. Hence, measures such as Cohen’s  $\kappa$  (1960) cannot be directly applied in this context because for crowdsourced annotations, we cannot meaningfully compute annotator-specific distribution of the labels and biases.

Second, most of the standard annotation metrics do not generalize for flat segmentation and trees. Artstein and Poesio (2008) provides a comprehensive survey of the IAA metrics and their usage in NLP. They note that all the metrics assume that a fixed set of labels are used for items. Therefore, it is far from obvious how to compare chunking or segmentation that *covers* the whole text or that might have *overlapping* units as in the case of nested segmentation. Furthermore, we would like to compare the reliability of flat and nested segmentation, and therefore, ideally we would like to have an IAA metric that can be meaningfully applied to both of these cases.

After considering various measures, we decided to appropriately generalize one of the most versatile and effective IAA metrics proposed till date, the Krippendorff’s  $\alpha$  (2004). To be consistent with prior work, we will stick to the notation used in Artstein and Poesio (2008) and redefine the  $\alpha$  in the context of flat and nested segmentation. Note that though the notations introduced here will be from the perspective of queries, it is equally applicable to sentences and the generalization is

straightforward.

### 4.1 Notations and Definitions

Let  $Q$  be the set of all queries with cardinality  $q$ . A query  $q \in Q$  can be represented as a sequence of  $|q|$  words:  $w_1 w_2 \dots w_{|q|}$ . We introduce  $|q| - 1$  random variables,  $b_1, b_2, \dots, b_{|q|-1}$ , such that  $b_i$  represents the boundary between the words  $w_i$  and  $w_{i+1}$ . A flat or nested segmentation of  $q$ , represented by  $q_j$ ,  $j$  varying from 1 to total number of annotations  $c$ , is a particular instantiation of these boundary variables as described below.

*Definition.* A *flat segmentation*,  $q_j$  can be uniquely defined by a binary assignment of the boundary variables  $b_{j,i}$ , where  $b_{j,i} = 1$  iff  $w_i$  and  $w_{i+1}$  belong to two different flat segments. Otherwise,  $b_{j,i} = 0$ . Thus,  $q$  has  $2^{|q|-1}$  possible flat segmentations.

*Definition.* A *nested segmentation*  $q_j$  can also be uniquely defined by assigning non-negative integers to the boundary variables such that  $b_{j,i} = 0$  iff words  $w_i$  and  $w_{i+1}$  form an atomic segment (i.e., they are grouped together), else  $b_{j,i} = 1 + \max(\text{left}_i, \text{right}_i)$ , where  $\text{left}_i$  and  $\text{right}_i$  are the heights of the largest subtrees ending at  $w_i$  and beginning at  $w_{i+1}$  respectively.

This numbering scheme for nested segmentation can be understood through Fig. 1. Every internal node of the binary tree corresponding to the nested segmentation is numbered according to its height. The lowest internal nodes, both of whose children are query words, are assigned a value of 0. Other internal nodes get a value of one greater than the height of its higher child. Since every internal node corresponds to a boundary, we assign the height of the node to the corresponding boundaries. The number of unique nested segmentations of a query of length  $|q|$  is its corresponding *Catalan number*<sup>7</sup>.

Boundary variables for flat and nested segmentation are illustrated with an example of each kind in Tables 1 and 2 (last column).

### 4.2 Krippendorff’s $\alpha$ for Segmentation

Krippendorff’s  $\alpha$  (Krippendorff, 2004) is an extremely versatile agreement coefficient, which is based on the assumption that the expected agreement is calculated by looking at the overall distribution of judgments without regard to which annotator produced them (Artstein and Poesio, 2008).

<sup>7</sup><http://goo.gl/vKQvK>

Hence, it is appropriate for crowdsourced annotation, where the judgments come from a large number of unrelated annotators. Moreover, it allows for different magnitudes of disagreement, which is a useful feature as we might want to differentially penalize disagreements at various levels of the tree for nested segmentation.

$\alpha$  is defined as

$$\alpha = 1 - \frac{D_o}{D_e} = 1 - \frac{s_{within}^2}{s_{total}^2} \quad (1)$$

where  $D_o$  and  $D_e$  are, respectively, the observed and expected disagreements that are measured by  $s_{within}^2$  – the variance within the annotation of an item and  $s_{total}^2$  – variance across annotations of all items. We adapt the equations presented in pp.565-566 of Artstein and Poesio (2008) for measuring these quantities for queries:

$$s_{within}^2 = \frac{1}{2\mathbf{q}\mathbf{c}(\mathbf{c}-1)} \sum_{q \in Q} \sum_{m=1}^{\mathbf{c}} \sum_{n=1}^{\mathbf{c}} d(q_m, q_n) \quad (2)$$

$$s_{total}^2 = \frac{1}{2\mathbf{q}\mathbf{c}(\mathbf{q}\mathbf{c}-1)} \sum_{q \in Q} \sum_{m=1}^{\mathbf{c}} \sum_{q' \in Q} \sum_{n=1}^{\mathbf{c}} d(q_m, q'_n) \quad (3)$$

where,  $d(q_m, q'_n)$  is a distance metric for the agreement between annotations  $q_m$  and  $q'_n$ .

We define two different distance metrics  $d_1$  and  $d_2$  that are applicable to flat and nested segmentation. We shall first define these metrics for comparing queries with equal length (i.e.,  $|q| = |q'|$ ):

$$d_1(q_m, q'_n) = \frac{1}{|q|-1} \sum_{i=1}^{|q|-1} |b_{m,i} - b'_{n,i}| \quad (4)$$

$$d_2(q_m, q'_n) = \frac{1}{|q|-1} \sum_{i=1}^{|q|-1} |b_{m,i}^2 - (b'_{n,i})^2| \quad (5)$$

While  $d_1$  penalizes all disagreements equally,  $d_2$  penalizes disagreements higher up the tree more.  $d_2$  might be a desirable metric for nested segmentation, because research on sentence chunking shows that annotators agree more on clause or major phrase boundaries, even though they may not always agree on intra-clausal or intra-phrasal boundaries (Bali et al., 2009). Note that for flat segmentation,  $d_1$  and  $d_2$  are identical, and hence we will denote them as  $d$ .

We propose the following extension to these metrics for queries of unequal lengths. Without

loss of generality, let us assume that  $|q| < |q'|$ .  $k$  is 1 or 2;  $r = |q'| - |q| + 1$ .

$$d_k(q_m, q'_n) = \frac{1}{r(|q|-1)} \sum_{a=0}^{r-1} \sum_{i=1}^{|q|-1} |b_{m,i}^k - (b'_{n,i+a})^k| \quad (6)$$

### 4.3 IAA under Random Bias Assumption

Krippendorff’s  $\alpha$  uses the cross-item variance as an estimate of chance agreement, which is reliable in general. However, this might result in misleadingly low values of IAA, especially when the items in the set are indeed expected to have similar annotations. To resolve this, we also compute the chance agreement under a random bias model. The random model assumes that *all the structural annotations of  $q$  are equiprobable*. For flat segmentation, it boils down to the fact that all the  $2^{|q|-1}$  annotations are equally likely, which is equivalent to the assumption that any boundary variable  $b_i$  has 0.5 probability of being 0 and 0.5 for 1.

Analytical computation of the expected probability distributions of  $d_1(q_m, q_n)$  and  $d_2(q_m, q_n)$  is harder for nested segmentation. Therefore, we programmatically generate all possible trees for  $q$ , which is again dependent only on  $|q|$  and compute  $d_1$  and  $d_2$  between all pairs of trees, from which the expected distributions can be readily estimated. Let us denote this expected cumulative probability distribution for flat segmentation as  $P_d(x; |q|)$  = the probability that for a pair of randomly chosen flat segmentations of  $q$ ,  $q_m$  and  $q_n$ ,  $d(q_m, q_n) \geq x$ . Likewise, let  $P_{d_1}(x; |q|)$  and  $P_{d_2}(x; |q|)$  be the respective probabilities that for any two nested segmentations  $q_m$  and  $q_n$  of  $q$ , the following holds:  $d_1(q_m, q_n) \geq x$  and  $d_2(q_m, q_n) \geq x$ .

We define the IAA under random bias model as ( $k$  is 1, 2 or null):

$$S = \frac{1}{\mathbf{q}\mathbf{c}^2} \sum_{q \in Q} \sum_{m=1}^{\mathbf{c}} \sum_{n=1}^{\mathbf{c}} P_{d_k}(d_k(q_m, q_n); |q|) \quad (7)$$

Thus,  $S$  is the expected probability of observing a similar or worse agreement by random chance, averaged over all pairs of annotations for all queries, and not a chance corrected IAA metric such as  $\alpha$ . Thus,  $S = 1$  implies that the observed agreement is *almost always better than* that by random chance and  $S = 0.5$  and 0 respectively imply that the observed agreement is *as good as* and *almost always worse than* that by random chance. We

Dataset	Flat	Nested	
	$d_1$	$d_1$	$d_2$
Q700	0.21(0.59)	0.21(0.89)	0.16(0.68)
Q500	0.22(0.62)	0.15(0.70)	0.15(0.44)
QG500	0.61(0.88)	0.66(0.88)	0.67(0.80)
S300	0.27(0.74)	0.18(0.94)	0.14(0.75)
U250	0.23(0.89)	0.42(0.90)	0.30(0.78)
B250	0.22(0.86)	0.34(0.88)	0.22(0.71)
T250	0.20(0.86)	0.44(0.89)	0.34(0.76)

Table 4: Agreement Statistics:  $\alpha(S)$ .

also note that a high value of  $S$  and low value of  $\alpha$  indicate that though the annotators agree on the judgment of individual items, they also tend to agree on judgments of two different items, which in turn, could be due to strong annotator biases or due to lack of variability of the dataset.

In the supplementary material, computations of  $\alpha$  and  $S$  have been explained in further details through worked out examples. Tables for the expected distributions of  $d$ ,  $d_1$  and  $d_2$  under the random annotation assumption are also available.

## 5 Results

Table 4 reports the values of  $\alpha$  and  $S$  for flat and nested segmentation on the various datasets. For nested segmentation, the values were computed for two different distance metrics  $d_1$  and  $d_2$ . As expected, the highest value of  $\alpha$  for both flat and nested segmentation is observed for gold annotations. An  $\alpha > 0.6$  indicates quite good IAA, and thus, reliable annotations. Higher  $\alpha$  for nested segmentation **QG500** than flat further validates our initial postulate that nested segmentation may reduce disagreement from granularity issues inherent in the definition of flat segmentation.

Opposite trends are observed for **Q700**, **Q500** and **S300**, where  $\alpha$  for flat is the highest, followed by that for nested using  $d_1$ , and then  $d_2$ . Moreover, except for flat segmentation of sentences,  $\alpha$  lies between 0.14 and 0.22, which is quite low. This clearly shows that segmentation, either flat or nested, cannot be reliably procured through crowdsourcing. Lower  $\alpha$  for  $d_2$  than  $d_1$  further indicates that annotators disagree more for higher levels of the trees, contrary to what we had expected. However, nearly equal IAA for sentences and queries implies that low agreement may not be an outcome of inherent ambiguity in the structure

of queries. Slightly higher  $\alpha$  for flat segmentation and a much higher  $\alpha$  for nested segmentation of **QRand** reinforce the fact that low IAA is not due to a lack of structure in queries.

It is interesting to note that  $\alpha$  for nested segmentation of **S300** and all segmentations of **QRand** are low or medium despite the fact that  $S$  is very high in all these cases. Thus, it is clear that annotators have a strong bias towards certain structures across queries. In the next section, we will analyze some of these biases. We also computed the IAA between **QG500** and **Q500**, and found  $\alpha = 0.27$ . This is much lower than  $\alpha$  for **QG500**, though slightly higher than that for **Q500**. We did not observe any significant variation in agreement with respect to the length of the queries.

## 6 Biases in Annotation

The IAA statistics clearly show that there are certain strong biases in both flat and nested query segmentation, especially those obtained through crowdsourcing. To identify these biases, we went through the annotations and came up with possible hypotheses, which we tried to verify through statistical analysis of the data. Here, we report the most prominent biases that were thus discovered.

**Bias 1:** *During flat segmentation, annotators prefer dividing the query into two segments of roughly equal length.*

As discussed earlier, one of the major problems of flat segmentation is the fuzziness in granularity. In our experiments, we intentionally left the decision of whether to go for fine or coarse-grained segmentation to the annotator. However, it is surprising to observe that annotators typically divide the query into two segments (see Fig. 3, plots A1 and A2), and at times three, but hardly ever more than three. This bias is observed across queries, sentences and random queries, where the percentage of annotations with 2 or 3 segments are greater than 83%, 91% and 96% respectively. This bias is most strongly visible for **QRand** because the lack of syntactic or semantic cohesion between the words provides no clue for segmentation.

Furthermore, we observe that typically segments tend to be of equal length. For this, we computed standard deviations (sd) of segment lengths for all annotations having 2 or 3 segments; the distribution of sd is shown in Fig. 3, plots B1 and B2. We observe that for all datasets, sd lies mainly between 0.5 and 1 (for perspective, consider a query

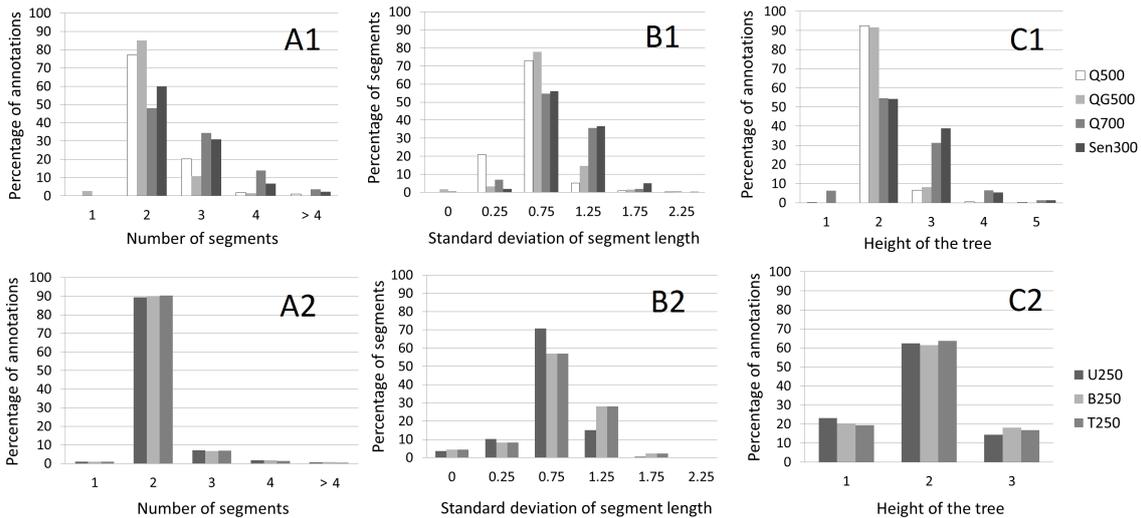


Figure 3: Analysis of annotation biases: A1, A2 – number of segments per flat segmentation vs. length; B1, B2 – standard deviation of segment length for flat segmentation; C1, C2 – distribution of the tree heights in nested segmentation.

Length	Expected	Q500	QG500	Q700	S300	QRand
5	2.57	2.00	2.02	2.08	2.02	2.01
6	3.24	2.26	2.23	2.23	2.24	2.02
7	3.88	2.70	2.71	2.67	2.55	2.62
8	4.47	2.89	2.68	2.72	2.72	2.35

Table 5: Average height for nested segmentation.

with 7 words; with two segments of length 3 and 4 the sd is 0.5, and for 2 and 5, the sd is 1.5), implying that segments are roughly of equal length.

It is likely that due to this bias, the  $S$  or observed agreement is moderately high for queries and very high for sentences, but then it also leads to high agreement across different queries and sentences (i.e., high  $s_{total}^2$ ) especially when they are of equal length, which in turn brings down the value of  $\alpha$  – the true agreement after bias correction.

**Bias 2:** *During nested segmentation, annotators prefer balanced binary trees.*

Quite analogous to bias 1, for nested segmentation we observe that annotators tend to prefer more balanced binary trees. Fig. 3 plots C1 and C2 show the distribution of the tree heights for various cases and Table 5 reports the corresponding average height of the trees for queries and sentences of various lengths and the the expected value of the height if all trees were equally likely. The observed heights are much lower than the expected values clearly implying the preference of the annotators for more balanced trees.

Thus, the crowd seems to choose the middle

path, avoiding extremes and hence may not be a reliable source of annotation for query segmentation. It can be argued that similar biases are also observed for gold annotations, and therefore, probably it is the inherent structure of the queries and sentences that lead to such biased distribution of segmentation patterns. However, note that  $\alpha$  for **QG500** is much higher than all other cases, which shows that the true agreement between gold annotators is immune to such biases or skewed distributions in the datasets. Furthermore, high values of  $\alpha$  for **QRand** despite the very strong biases in annotation shows that there perhaps is very little choice that the annotators have while segmenting randomly generated queries. On the other hand, the textual coherence of the real queries and sentences provide many different choices for segmentation and the Turker typically gets carried away by these biases, leading to low  $\alpha$ .

**Bias 3:** *Phrase structure drives segmentation only when reconcilable with Bias 1.* Whenever the sentence or query has a verb phrase (VP) spanning roughly half of it, annotators seem to chunk before the VP as one would expect, quite as often as just after the verb, which is quite unexpected. For instance, the sentence `A gentle sarcasm ruffled her anger. gathers as many as eight flat annotations with a boundary between sarcasm and ruffled, and four with a boundary between ruffled and her.` However, if the VP is very short consisting of a single

Position	Q500	QG500	Q700	S300	QRand
Both	2.24	0.37	2.78	2.08	0.63
None	50.34	56.85	35.74	35.84	39.81
Right	23.86	21.50	19.02	12.52	15.23
Left	18.08	15.97	40.59	45.96	21.21

Table 6: Percentages of positions of segment boundaries with respect to prepositions. Prepositions occurring in the beginning or end of a query/sentence have been excluded from the analysis; hence, numbers in a column do not total 100.

verb, as in `A fleeting and furtive air of triumph erupted.`, annotators seem to attempt for a balanced annotation due to **Bias 1**. As a clear middle boundary is not present in such sentences, the annotations show a lot more variation and disagreement. For instance, only 1 out of 10 annotations had a boundary before `erupted` in the above example. In fact, at least one annotation had a boundary after each word in the sentence, with no clear majority.

**Bias 4:** *Prepositions influence segment boundaries differently for queries and sentences.* We automatically labeled all the prepositions in the flat annotations and classified them according to the criterion of whether a boundary was placed immediately before or after it, or on both sides or neither side. The statistics, reported in Table 6, show that for NL sentences a majority of the boundaries are present before the preposition, marking the beginning of a prepositional phrase. However, for queries, a much richer pattern emerges depending on the specific preposition. For instance, `to`, `of` and `for` are often chunked with the previous word (e.g., `how to | choose a bike size, birthday party ideas for | one year old`). We believe that this difference is because in sentences due to the presence of a verb, the PP has a well-defined head, lack of which leads to preposition in queries getting chunked with words that form more commonly seen patterns (e.g., `flights to and tickets for`).

Bias 3 and 4 present the complex interpretation of the structure of queries by the annotators which could be due to some emerging cognitive model of queries among the search engine users. This is a fascinating and unexplored aspect of query structures that demands deeper investigation through cognitive and psycholinguistic experiments.

## 7 Conclusion

We have studied various aspects of query segmentation through crowdsourcing by designing and conducting suitable experiments. Analysis of experimental data leads us to conclude the following: (a) crowdsourcing may not be a very effective way to collect judgments for query segmentation; (b) addressing fuzziness of granularity for flat segmentation by introducing strict binary nested segments does not lead to better agreement in crowdsourced annotations, though it definitely improves the IAA for gold standard segmentations, implying that low IAA in flat segmentation among experts is primarily an effect of unspecified granularity of segments; (c) low IAA is not due to the inherent structural ambiguity in queries as this holds true for sentences as well; (d) there are strong biases in crowdsourced annotations, mostly because turkers prefer more balanced segment structures; and (e) while annotators are by and large guided by linguistic principles, application of these principles differ between query and NL sentences and also closely interact with other biases.

One of the important contributions of this work is the formulation of a new IAA metric for comparing across flat and nested segmentations, especially for crowdsourcing based annotations. Since trees are commonly used across various linguistic annotations, this metric can have wide applicability. The metric, moreover, can be easily adapted to other annotation schemes as well by defining an appropriate distance metric between annotations. Since large scale data for query segmentation is very useful, it would be interesting to see if the problem can be rephrased to the Turkers in a way so as to obtain more reliable judgments. Yet a deeper question is regarding the theoretical status of query structure, which though in an emergent state is definitely an operating model for the annotators. Our future work in this area would specifically target understanding and formalization of the theoretical model underpinning a query.

## Acknowledgments

We thank Ed Cutrell and Andrew Cross, Microsoft Research Lab India, for their help in setting up the AMT experiments. We would also like to thank Anusha Suresh, IIT Kharagpur, India, for helping us with data preparation.

## References

- Steven P. Abney. 1991. *Parsing By Chunks*. Kluwer Academic Publishers.
- Steven P. Abney. 1992. Prosodic Structure, Performance Structure And Phrase Structure. In *Proceedings 5th DARPA Workshop on Speech and Natural Language*, pages 425–428. Morgan Kaufmann.
- Steven P. Abney. 1995. Chunks and dependencies: Bringing processing evidence to bear on syntax. *Computational Linguistics and the Foundations of Linguistic Theory*, pages 145–164.
- Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.
- Kalika Bali, Monojit Choudhury, Diptesh Chatterjee, Sankalan Prasad, and Arpit Maheswari. 2009. Correlates between Performance, Prosodic and Phrase Structures in Bangla and Hindi: Insights from a Psycholinguistic Experiment. In *Proceedings of International Conference on Natural Language Processing*, pages 101 – 110.
- Michael Bendersky, W. B. Croft, and David A. Smith. 2009. Two-stage query segmentation for information retrieval. In *Proceedings of the 32nd international ACM Special Interest Group on Information Retrieval (SIGIR) Conference on Research and Development in Information Retrieval*, pages 810–811. ACM.
- Shane Bergsma and Qin Iris Wang. 2007. Learning Noun Phrase Query Segmentation. In *Proceedings of Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 819–826.
- Akshar Bharati, Vineet Chaitanya, Rajeev Sangal, and KV Ramakrishnamacharyulu. 1995. *Natural language processing: a Paninian perspective*. Prentice-Hall of India New Delhi.
- David J. Brenes, Daniel Gayo-Avello, and Rodrigo Garcia. 2010. On the fly query segmentation using snippets. In *CERI '10*, pages 259–266.
- Chris Callison-Burch. 2009. Fast, cheap, and creative: evaluating translation quality using amazon’s mechanical turk. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP '09*, pages 286–295. Association for Computational Linguistics.
- Vitor R Carvalho, Matthew Lease, and Emine Yilmaz. 2011. Crowdsourcing for search evaluation. *ACM Sigir forum*, 44(2):17–22.
- Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.
- Zhicheng Dou, Ruihua Song, Xiaojie Yuan, and Ji-Rong Wen. 2008. Are Click-through Data Adequate for Learning Web Search Rankings? In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 73–82. ACM.
- Matthias Hagen, Martin Potthast, Benno Stein, and Christof Bräutigam. 2011. Query Segmentation Revisited. In *Proceedings of the 20th International Conference on World Wide Web*, pages 97–106. ACM.
- Matthias Hagen, Martin Potthast, Anna Beyer, and Benno Stein. 2012. Towards Optimum Query Segmentation: In Doubt Without. In *Proceedings of the Conference on Information and Knowledge Management*, pages 1015–1024.
- Jian Huang, Jianfeng Gao, Jiangbo Miao, Xiaolong Li, Kuansan Wang, Fritz Behr, and C. Lee Giles. 2010. Exploring web scale language models for search query processing. In *Proceedings of the 19th international conference on World wide web, WWW '10*, pages 451–460, New York, NY, USA. ACM.
- Klaus Krippendorff. 2004. *Content Analysis: An Introduction to its Methodology*. Sage, Thousand Oaks, CA.
- Yanen Li, Bo-Jun Paul Hsu, ChengXiang Zhai, and Kuansan Wang. 2011. Unsupervised query segmentation using clickthrough for information retrieval. In *SIGIR '11*, pages 285–294. ACM.
- Knut Magne Risvik, Tomasz Mikolajewski, and Peter Boros. 2003. Query segmentation for web search. In *WWW (Posters)*.
- Rishiraj Saha Roy, Niloy Ganguly, Monojit Choudhury, and Srivatsan Laxman. 2012. An IR-based Evaluation Framework for Web Search Query Segmentation. In *Proceedings of the International ACM Special Interest Group on Information Retrieval (SIGIR) Conference on Research and Development in Information Retrieval*, pages 881–890. ACM.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 254–263, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Bin Tan and Fuchun Peng. 2008. Unsupervised Query Segmentation Using Generative Language Models and Wikipedia. In *Proceedings of the 17th International Conference on World Wide Web (WWW)*, pages 347–356. ACM.
- Chao Zhang, Nan Sun, Xia Hu, Tingzhu Huang, and Tat-Seng Chua. 2009. Query segmentation based on eigenspace similarity. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, ACLShort '09*, pages 185–188, Stroudsburg, PA, USA. Association for Computational Linguistics.

# Deceptive Answer Prediction with User Preference Graph

Fangtao Li<sup>§</sup>, Yang Gao<sup>†</sup>, Shuchang Zhou<sup>§†</sup>, Xiance Si<sup>§</sup>, and Decheng Dai<sup>§</sup>

<sup>§</sup>Google Research, Mountain View

<sup>†</sup>State Key Laboratory of Computer Architecture, Institute of Computing Technology, CAS  
{lifangtao, georgezhou, sxc, decheng}@google.com

<sup>†</sup>Department of Computer Science and Technology, Tsinghua University  
gao\_young@163.com

## Abstract

In Community question answering (QA) sites, malicious users may provide deceptive answers to promote their products or services. It is important to identify and filter out these deceptive answers. In this paper, we first solve this problem with the traditional supervised learning methods. Two kinds of features, including textual and contextual features, are investigated for this task. We further propose to exploit the user relationships to identify the deceptive answers, based on the hypothesis that similar users will have similar behaviors to post deceptive or authentic answers. To measure the user similarity, we propose a new user preference graph based on the answer preference expressed by users, such as “helpful” voting and “best answer” selection. The user preference graph is incorporated into traditional supervised learning framework with the graph regularization technique. The experiment results demonstrate that the user preference graph can indeed help improve the performance of deceptive answer prediction.

## 1 Introduction

Currently, Community QA sites, such as Yahoo! Answers<sup>1</sup> and WikiAnswers<sup>2</sup>, have become one of the most important information acquisition methods. In addition to the general-purpose web search engines, the Community QA sites have emerged as popular, and often effective, means of information seeking on the web. By posting questions for other participants to answer, users can obtain answers to their specific questions. The Community QA

sites are growing rapidly in popularity. Currently there are hundreds of millions of answers and millions of questions accumulated on the Community QA sites. These resources of past questions and answers are proving to be a valuable knowledge base. From the Community QA sites, users can directly get the answers to meet some specific information need, rather than browse the list of returned documents to find the answers. Hence, in recent years, knowledge mining in Community QA sites has become a popular topic in the field of artificial intelligence (Adamic et al., 2008; Wei et al., 2011).

However, some answers may be deceptive. In the Community QA sites, there are millions of users each day. As the answers can guide the user’s behavior, some malicious users are motivated to give deceptive answers to promote their products or services. For example, if someone asks for recommendations about restaurants in the Community QA site, the malicious user may post a deceptive answer to promote the target restaurant. Indeed, because of lucrative financial rewards, in several Community QA sites, some business owners provide incentives for users to post deceptive answers for product promotion.

There are at least two major problems that the deceptive answers cause. On the user side, the deceptive answers are misleading to users. If the users rely on the deceptive answers, they will make the wrong decisions. Or even worse, the promoted link may lead to illegitimate products. On the Community QA side, the deceptive answers will hurt the health of the Community QA sites. A Community QA site without control of deceptive answers could only benefit spammers but could not help askers at all. If the asker was cheated by the provided answers, he will not trust and visit this site again. Therefore, it is a fundamental task to predict and filter out the deceptive answers.

In this paper, we propose to predict deceptive

<sup>1</sup><http://answers.yahoo.com>

<sup>2</sup><http://wiki.answers.com>

answer, which is defined as the answer, whose purpose is not only to answer the question, but also to promote the authors' self-interest. In the first step, we consider the deceptive answer prediction as a general binary-classification task. We extract two types of features: one is textual features from answer content, including unigram/bigram, URL, phone number, email, and answer length; the other is contextual features from the answer context, including the relevance between answer and the corresponding question, the author of the answer, answer evaluation from other users and duplication with other answers. We further investigate the user relationship for deceptive answer prediction. We assume that similar users tend to have similar behaviors, i.e. posting deceptive answers or posting authentic answers. To measure the user relationship, we propose a new user preference graph, which is constructed based on the answer evaluation expressed by users, such as "helpful" voting and "best answer" selection. The user preference graph is incorporated into traditional supervised learning framework with graph regularization, which can make answers, from users with same preference, tend to have the same category (deceptive or authentic). The experiment results demonstrate that the user preference graph can further help improve the performance for deceptive answer prediction.

## 2 Related Work

In the past few years, it has become a popular task to mine knowledge from the Community QA sites. Various studies, including retrieving the accumulated question-answer pairs to find the related answer for a new question, finding the expert in a specific domain, summarizing single or multiple answers to provide a concise result, are conducted in the Community QA sites (Jeon et al., 2005; Adamic et al., 2008; Liu et al., 2008; Song et al., 2008; Si et al., 2010a; Figueroa and Atkinson, 2011). However, an important issue which has been neglected so far is the detection of deceptive answers. If the acquired question-answer corpus contains many deceptive answers, it would be meaningless to perform further knowledge mining tasks. Therefore, as the first step, we need to predict and filter out the deceptive answers. Among previous work, answer quality prediction (Song et al., 2010; Harper et al., 2008; Shah and Pomerantz, 2010; Ishikawa et al., 2010) is most related to the deceptive answer prediction task. But these are

still significant differences between two tasks. Answer quality prediction measures the overall quality of the answers, which refers to the accuracy, readability, completeness of the answer. While the deceptive answer prediction aims to predict if the main purpose of the provided answer is only to answer the specific question, or includes the user's self-interest to promote something. Some of the previous work (Song et al., 2010; Ishikawa et al., 2010; Bian et al., 2009) views the "best answer" as high quality answers, which are selected by the askers in the Community QA sites. However, the deceptive answer may be selected as high-quality answer by the spammer, or because the general users are misled. Meanwhile, some answers from non-native speakers may have linguistic errors, which are low-quality answers, but are still authentic answers. Our experiments also show that answer quality prediction is much different from deceptive answer prediction.

Previous QA studies also analyze the user graph to investigate the user relationship (Jurczyk and Agichtein, 2007; Liu et al., 2011). They mainly construct the user graph with asker-answerer relationship to estimate the expertise score in Community QA sites. They assume the answerer is more knowledgeable than the asker. However, we don't care which user is more knowledgeable, but are more likely to know if two users are both spammers or authentic users. In this paper, we propose a novel user preference graph based on their preference towards the target answers. We assume that the spammers may collaboratively promote the target deceptive answers, while the authentic users may generally promote the authentic answers and demote the deceptive answers. The user preference graph is constructed based on their answer evaluation, such as "helpful" voting or "best answer" selection.

## 3 Proposed Features

We first view the deceptive answer prediction as a binary-classification problem. Two kinds of features, including textual features and contextual features, are described as follows:

### 3.1 Textual Features

We first aim to predict the deceptive answer by analyzing the answer content. Several textual features are extracted from the answer content:

#### 3.1.1 Unigrams and Bigrams

The most common type of feature for text classification is the bag-of-words. We use an effective

feature selection method  $\chi^2$  (Yang and Pedersen, 1997) to select the top 200 unigrams and bigrams as features. The top ten unigrams related to deceptive answers are shown on Table 1. We can see that these words are related to the intent for promotion.

professional	service	advice	address
site	telephone	therapy	recommend
hospital	expert		

Table 1: Top 10 Deceptive Related Unigrams

### 3.1.2 URL Features

Some malicious users may promote their products by linking a URL. We find that URL is good indicator for deceptive answers. However, some URLs may provide the references for the authentic answers. For example, if you ask the weather in mountain view, someone may just post the link to "http://www.weather.com/". Therefore, besides the existence of URL, we also use the following URL features:

- 1). Length of the URLs: we observe that the longer urls are more likely to be spam.
- 2). PageRank Score: We employ the PageRank (Page et al., 1999) score of each URL as popularity score.

### 3.1.3 Phone Numbers and Emails

There are a lot of contact information mentioned in the Community QA sites, such as phone numbers and email addresses, which are very likely to be deceptive, as good answers are found to be less likely to refer to phone numbers or email addresses than the malicious ones. We extract the number of occurrences of email and phone numbers as features.

### 3.1.4 Length

We have also observed some interesting patterns about the length of answer. Deceptive ones tend to be longer than authentic ones. This can be explained as the deceptive answers may be well prepared to promote the target. We also employ the number of words and sentences in the answer as features.

## 3.2 Contextual Features

Besides the answer textual features, we further investigate various features from the context of the target answer:

### 3.2.1 Question Answer Relevance

The main characteristic of answer in Community QA site is that the answer is provided to answer the corresponding question. We can use the corresponding question as one of the context features by measuring the relevance between the answer and the question. We employ three different models for Question-Answer relevance:

#### Vector Space Model

Each answer or question is viewed as a word vector. Given a question  $q$  and the answer  $a$ , our vector model uses weighted word counts(e.g.TF-IDF) as well as the cosine similarity ( $q \cdot a$ ) of their word vectors as relevant function (Salton and McGill, 1986). However, vector model only consider the exact word match, which is a big problem, especially when the question and answer are generally short compared to the document. For example, Barack Obama and the president of the US are the same person. But the vector model would indicate them to be different. To remedy the word-mismatch problem, we also look for the relevance models in higher semantic levels.

#### Translation Model

A translation model is a mathematical model in which the language translation is modeled in a statistical way. The probability of translating a source sentence (as answer here) into target sentence (as question here) is obtained by aligning the words to maximize the product of all the word probabilities. We train a translation model (Brown et al., 1990; Och and Ney, 2003) using the Community QA data, with the question as the target language, and the corresponding best answer as the source language. With translation model, we can compute the translation score for new question and answer.

#### Topic Model

To reduce the false negatives of word mismatch in vector model, we also use the topic models to extend matching to semantic topic level. The topic model, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003), considers a collection of documents with  $K$  latent topics, where  $K$  is much smaller than the number of words. In essence, LDA maps information from the word dimension to a semantic topic dimension, to address the shortcomings of the vector model.

### 3.2.2 User Profile Features

We extract several user's activity statistics to construct the user profile features, including the level

of the user in the Community QA site, the number of questions asked by this user, the number of answers provided by this user, and the best answer ratio of this user.

### 3.2.3 User Authority Score

Motivated by expert finding task (Jurczyk and Agichtein, 2007; Si et al., 2010a; Li et al., 2011), the second type of author related feature is authority score, which denotes the expertise score of this user. To compute the authority score, we first construct a directed user graph with the user interactions in the community. The nodes of the graph represent users. An edge between two users indicates a contribution from one user to the other. Specifically, on a Q&A site, an edge from A to B is established when user B answered a question asked by A, which shows user B is more likely to be an expert than A. The weight of an edge indicates the number of interactions. We compute the user’s authority score (AS) based on the link analysis algorithm *PageRank*:

$$AS(u_i) = \frac{1-d}{N} + d \sum_{u_j \in M(u_i)} \frac{AS(u_j)}{L(u_j)} \quad (1)$$

where  $u_1, \dots, u_N$  are the users in the collection,  $N$  is the total number of users,  $M(u_i)$  is the set of users whose answers are provided by user  $u_i$ ,  $L(u_i)$  is the number of users who answer  $u_i$ ’s questions,  $d$  is a damping factor, which is set as 0.85. The authority score can be computed iteratively with random initial values.

### 3.2.4 Robot Features

The third type of author related feature is used for detecting whether the author is a robot, which are scripts crafted by malicious users to automatically post answers. We observe that the distributions of the answer-posting time are very different between general user and robot. For example, some robots may make posts continuously and mechanically, hence the time increment may be smaller than human users who would need time to think and process between two posts. Based on this observation, we design a time sequence feature for robot detection. For each author, we can get a list of time points to post answers,  $T = \{t_0, t_1, \dots, t_n\}$ , where  $t_i$  is the time point when posting the  $i$ th answer. We first convert the time sequence  $T$  to time interval sequence  $\Delta T = \{\Delta t_0, \Delta t_1, \dots, \Delta t_{n-1}\}$ , where  $\Delta t_i = t_{i+1} - t_i$ . Based on the interval sequences for all users, we then construct a matrix  $X_{m \times b}$  whose rows correspond to users and

columns correspond to interval histogram with predefined range. We can use each row vector as time sequence pattern to detect robot. To reduce the noise and sparse problem, we use the dimension reduction techniques to extract the latent semantic features with Singular Value Decomposition (SVD) (Deerwester et al., 1990; Kim et al., 2006).

### 3.2.5 Evaluation from Other Users

In the Community QA sites, other users can express their opinions or evaluations on the answer. For example, the asker can choose one of the answers as best answer. We use a bool feature to denote if this answer is selected as the best answer. In addition, other users can label each answer as “helpful” or “not helpful”. We also use this helpful evaluation by other users as the contextual feature, which is defined as the ratio between the number of “helpful” votes and the number of total votes.

### 3.2.6 Duplication with Other Answers

The malicious user may post the pre-written product promotion documents to many answers, or just change the product name. We also compute the similarity between different answers. If the two answers are totally same, but the question is different, these answer is potentially as a deceptive answer. Here, we don’t want to measure the semantic similarity between two answers, but just measure if two answers are similar to the word level, therefore, we apply BleuScore (Papineni et al., 2002), which is a standard metric in machine translation for measuring the overlap between n-grams of two text fragments  $r$  and  $c$ . The duplication score of each answer is the maximum BleuScore compared to all other answers.

## 4 Deceptive Answer Prediction with User Preference Graph

Besides the textual and contextual features, we also investigate the user relationship for deceptive answer prediction. We assume that similar users tend to perform similar behaviors (posting deceptive answers or posting authentic answers). In this section, we first show how to compute the user similarity (user preference graph construction), and then introduce how to employ the user relationship for deceptive answer prediction.

### 4.1 User Preference Graph Construction

In this section, we propose a new user graph to describe the relationship among users. Figure 1 (a) shows the general process in a question answering

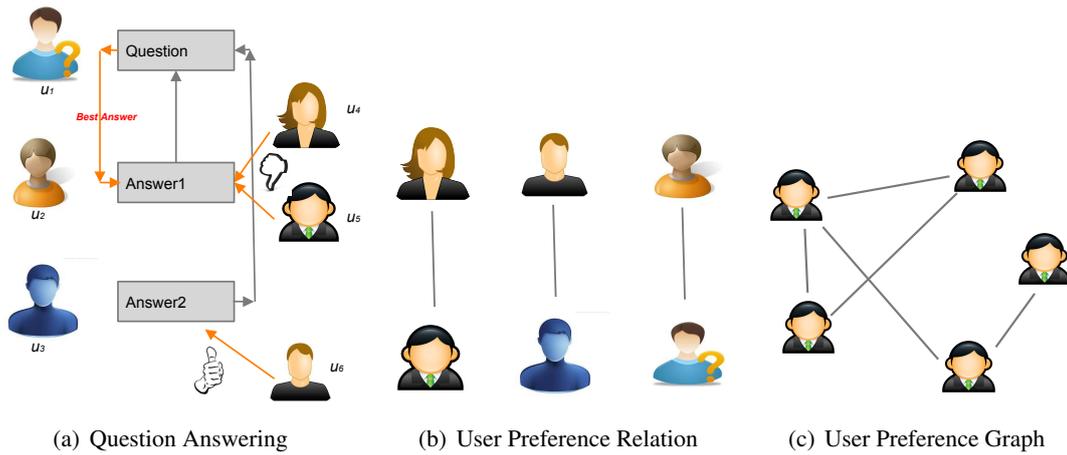


Figure 1: User Preference Graph Construction

thread. The asker, i.e.  $u_1$ , asks a question. Then, there will be several answers to answer this question from other users, for example, answerers  $u_2$  and  $u_3$ . After the answers are provided, users can also vote each answer as “helpful” or “not helpful” to show their evaluation towards the answer. For example, users  $u_4, u_5$  vote the first answer as “not helpful”, and user  $u_6$  votes the second answer as “helpful”. Finally, the asker will select one answer as the best answer among all answers. For example, the asker  $u_1$  selects the first answer as the “best answer”.

To mine the relationship among users, previous studies mainly focus on the asker-answerer relationship (Jurczyk and Agichtein, 2007; Liu et al., 2011). They assume the answerer is more knowledgeable than the asker. Based on this assumption, they can extract the expert in the community, as discussed in Section 3.2.3. However, we don’t care which user is more knowledgeable, but are more interested in whether two users are both malicious users or authentic users. Here, we propose a new user graph based on the user preference. The preference is defined based on the answer evaluation. If two users show same preference towards the target answer, they will have the user-preference relationship. We mainly use two kinds of information: “helpful” evaluation and “best answer” selection. If two users give same “helpful” or “not helpful” to the target answer, we view these two users have same user preference. For example, user  $u_4$  and user  $u_5$  both give “not helpful” evaluation towards the first answer, we can say that they have same user preference. Besides the real “helpful” evaluation, we also assume the author of the answer gives the “helpful” evalu-

ation to his or her own answer. Then if user  $u_6$  give “helpful” evaluation to the second answer, we will view user  $u_6$  has same preference as user  $u_3$ , who is the author of the second answer. We also can extract the user preference with “best answer” selection. If the asker selects the “best answer” among all answers, we will view that the asker has same preference as the author of the “best answer”. For example, we will view user  $u_1$  and user  $u_2$  have same preference.

Based on the two above assumptions, we can extract three user preference relationships (with same preference) from the question answering example in Figure 1 (a):  $u_4 \sim u_5, u_3 \sim u_6, u_1 \sim u_2$ , as shown in Figure 1 (b). After extracting all user preference relationships, we can construct the user preference graph as shown in Figure 1 (c). Each node represents a user. If two users have the user preference relationship, there will be an edge between them. The edge weight is the number of user preference relationships.

In the Community QA sites, the spammers mainly promote their target products by promoting the deceptive answers. The spammers can collaboratively make the deceptive answers look good, by voting them as high-quality answer, or selecting them as “best answer”. However, the authentic users generally have their own judgements to the good and bad answers. Therefore, the evaluation towards the answer reflects the relationship among users. Although there maybe noisy relationship, for example, an authentic user may be cheated, and selects the deceptive answer as “best answer”, we hope the overall user preference relation can perform better results than previous user interaction graph for this task.

## 4.2 Incorporating User Preference Graph

To use the user graph, we can just compute the feature value from the graph, and add it into the supervised method as the features introduced in Section 3. Here, we propose a new technique to employ the user preference graph. We utilize the graph regularizer (Zhang et al., 2006; Lu et al., 2010) to constrain the supervised parameter learning. We will introduce this technique based on a commonly used model  $f(\cdot)$ , the linear weight model, where the function value is determined by linear combination of the input features:

$$f(\mathbf{x}_i) = \mathbf{w}^T \cdot \mathbf{x}_i = \sum_k w_k \cdot x_{ik} \quad (2)$$

where  $\mathbf{x}_i$  is a  $K$  dimension feature vector for the  $i$ th answer, the parameter value  $w_k$  captures the effect of the  $k$ th feature in predicting the deceptive answer. The best parameters  $\mathbf{w}^*$  can be found by minimizing the following objective function:

$$\Omega_1(\mathbf{w}) = \sum_i L(\mathbf{w}^T \mathbf{x}_i, y_i) + \alpha \cdot |\mathbf{w}|_F^2 \quad (3)$$

where  $L(\mathbf{w}^T \mathbf{x}_i, y_i)$  is a loss function that measures discrepancy between the predicted label  $\mathbf{w}^T \cdot \mathbf{x}_i$  and the true label  $y_i$ , where  $y_i \in \{+1, -1\}$ . The common used loss functions include  $L(p, y) = (p - y)^2$  (least square),  $L(p, y) = \ln(1 + \exp(-py))$  (logistic regression). For simplicity, here we use the least square loss function.  $|\mathbf{w}|_F^2 = \sum_k w_k^2$  is a regularization term defined in terms of the Frobenius norm of the parameter vector  $\mathbf{w}$  and plays the role of penalizing overly complex models in order to avoid fitting.

We want to incorporate the user preference relationship into the supervised learning framework. The hypothesis is that similar users tend to have similar behaviors, i.e. posting deceptive answers or authentic answers. Here, we employ the user preference graph to denote the user relationship. Based on this intuition, we propose to incorporate the user graph into the linear weight model with graph regularization. The new objective function is changed as:

$$\Omega_2(\mathbf{w}) = \sum_i L(\mathbf{w}^T \mathbf{x}_i, y_i) + \alpha \cdot |\mathbf{w}|_F^2 + \beta \sum_{u_i, u_j \in N_u} \sum_{x \in A_{u_i}, y \in A_{u_j}} w_{u_i, u_j} (f(x) - f(y))^2 \quad (4)$$

where  $N_u$  is the set of neighboring user pairs in user preference graph, i.e. the user pairs with same

preference.  $A_{u_i}$  is the set of all answers posted by user  $u_i$ .  $w_{u_i, u_j}$  is the weight of edge between  $u_i$  and  $u_j$  in user preference graph. In the above objective function, we impose a user graph regularization term

$$\beta \sum_{u_i, u_j \in N_u} \sum_{x \in A_{u_i}, y \in A_{u_j}} w_{u_i, u_j} (f(x) - f(y))^2$$

to minimize the answer authenticity difference among users with same preference. This regularization term smoothes the labels on the graph structure, where adjacent users with same preference tend to post answers with same label.

## 5 Experiments

### 5.1 Experiment Setting

#### 5.1.1 Dataset Construction

In this paper, we employ the Confucius (Si et al., 2010b) data to construct the deceptive answer dataset. Confucius is a community question answering site, developed by Google. We first crawled about 10 million question threads within a time range. Among these data, we further sample a small data set, and ask three trained annotators to manually label the answer as deceptive or not. If two or more people annotate the answer as deceptive, we will extract this answer as a deceptive answer. In total, 12446 answers are marked as deceptive answers. Similarly, we also manually annotate 12446 authentic answers. Finally, we get 24892 answers with deceptive and authentic labels as our dataset. With our labeled data, we employ supervised methods to predict deceptive answers. We conduct 5-fold cross-validation for experiments. The larger question threads data is employed for feature learning, such as translation model, and topic model training.

#### 5.1.2 Evaluation Metrics

The evaluation metrics are *precision*, *recall* and *F-score* for authentic answer category and deceptive answer category:  $precision = \frac{S_p \cap S_c}{S_p}$ ,  $recall = \frac{S_p \cap S_c}{S_c}$ , and  $F = \frac{2 * precision * recall}{precision + recall}$ , where  $S_c$  is the set of gold-standard positive instances for the target category,  $S_p$  is the set of predicted results. We also use the *accuracy* as one metric, which is computed as the number of answers predicted correctly, divided by the number of total answers.

	Deceptive Answer			Authentic Answer			Overall
	Prec.	Rec.	F-Score	Prec.	Rec.	F-Score	Acc.
Random	0.50	0.50	0.50	0.50	0.50	0.50	0.50
Unigram/Bigram (UB)	0.61	0.71	0.66	0.66	0.55	0.60	0.63
URL	0.93	0.26	0.40	0.57	0.98	0.72	0.62
Phone/Mail	0.94	0.15	0.25	0.53	0.99	0.70	0.57
Length	0.56	0.91	0.69	0.76	0.28	0.41	0.60
All Textual Features	0.64	0.67	0.66	0.66	0.63	0.64	0.65
QA Relevance	0.66	0.57	0.61	0.62	0.71	0.66	0.64
User Profile	0.62	0.53	0.57	0.59	0.67	0.63	0.60
User Authority	0.54	0.80	0.65	0.62	0.33	0.43	0.56
Robot	0.66	0.62	0.64	0.61	0.66	0.64	0.64
Answer Evaluation	0.55	0.53	0.54	0.55	0.57	0.56	0.55
Answer Duplication	0.69	0.71	0.70	0.70	0.68	0.69	0.69
All Contextual Feature	0.78	0.74	0.76	0.75	0.79	0.77	0.77
Textual + Contextual	0.80	0.82	0.81	0.82	0.79	0.80	0.81

Table 2: Results With Textual and Contextual Features

## 5.2 Results with Textual and Contextual Features

We tried several different classifiers, including SVM, ME and the linear weight models with least square and logistic regression. We find that they can achieve similar results. For simplicity, the linear weight with least square is employed in our experiment. Table 2 shows the experiment results. For textual features, it achieves much better result with unigram/bigram features than the random guess. This is very different from the answer quality prediction task. The previous studies (Jeon et al., 2006; Song et al., 2010) find that the word features can’t improve the performance on answer quality prediction. However, from Table 1, we can see that the word features can provide some weak signals for deceptive answer prediction, for example, words “recommend”, “address”, “professional” express some kinds of promotion intent. Besides unigram and bigram, the most effective textual feature is URL. The phone and email features perform similar results with URL. The observation of length feature for deceptive answer prediction is very different from previous answer quality prediction. For answer quality prediction, length is an effective feature, for example, long-length provides very strong signals for high-quality answer (Shah and Pomerantz, 2010; Song et al., 2010). However, for deceptive answer prediction, we find that the long answers are more potential to be deceptive. This is because most of deceptive answers are well prepared for product

promotion. They will write detailed answers to attract user’s attention and promote their products. Finally, with all textual features, the experiment achieves the best result, 0.65 in accuracy.

For contextual features, we can see that, the most effective contextual feature is answer duplication. The malicious users may copy the prepared deceptive answers or just simply edit the target name to answer different questions. Question-answer relevance and robot are the second most useful single features for deceptive answer prediction. The main characteristics of the Community QA sites is to accumulate the answers for the target questions. Therefore, all the answers should be relevant to the question. If the answer is not relevant to the corresponding question, this answer is more likely to be deceptive. Robot is one of main sources for deceptive answers. It automatically post the deceptive answers to target questions. Here, we formulate the time series as interval sequence. The experiment result shows that the robot indeed has his own posting behavior patterns. The user profile feature also can contribute a lot to deceptive answer prediction. Among the user profile features, the user level in the Community QA site is a good indicator. The other two contextual features, including user authority and answer evaluation, provide limited improvement. We find the following reasons: First, some malicious users post answers to various questions for product promotion, but don’t ask any question. From Equation 1, when iteratively computing the

	Deceptive Answer			Authentic Answer			Overall
	Prec.	Rec.	F-Score	Prec.	Rec.	F-Score	Acc.
Interaction Graph as Feature	0.80	0.82	0.81	0.82	0.79	0.80	0.81
Interaction Graph as Regularizer	0.80	0.83	0.82	0.82	0.80	0.81	0.82
Preference Graph as Feature	0.79	0.83	0.81	0.82	0.78	0.80	0.81
Preference Graph as Regularizer	0.83	0.86	0.85	0.85	0.83	0.84	0.85

Table 3: Results With User Preference Graph

final scores, the authority scores for these malicious users will be accumulated to large values. Therefore, it is hard to distinguish whether the high authority score represents real expert or malicious user. Second, the “best answer” is not a good signal for deceptive answer prediction. This may be selected by malicious users, or the authentic asker was misled, and chose the deceptive answer as “best answer”. This also demonstrates that the deceptive answer prediction is very different from the answer quality prediction. When combining all the contextual features, it can achieve the overall accuracy 0.77, which is much better than the textual features. Finally, with all the textual and contextual features, we achieve the overall result, 0.81 in accuracy.

### 5.3 Results with User Preference Graph

Table 3 shows the results with user preference graph. We compare with several baselines. Interaction graph is constructed by the asker-answerer relationship introduced in Section 3.2.3. When using the user graph as feature, we compute the authority score for each user with PageRank as shown in Equation 1. We also incorporating the interaction graph with a regularizer as shown in Equation 4. Note that we didn’t consider the edge direction when using interaction graph as a regularizer. From the table, we can see that when incorporating user preference graph as a feature, it can’t achieve a better result than the interaction graph. The reason is similar as the interaction graph. The higher authority score may be boosted by other spammer, and can’t be a good indicator to distinguish deceptive and authentic answers. When we incorporate the user preference graph as a regularizer, it can achieve about 4% further improvement, which demonstrates that the user evaluation towards answers, such as “helpful” voting and “best answer” selection, is a good signal to generate user relationship for deceptive answer prediction, and the graph regularization is an effective technique to incorporate the user prefer-

ence graph. We also analyze the parameter sen-

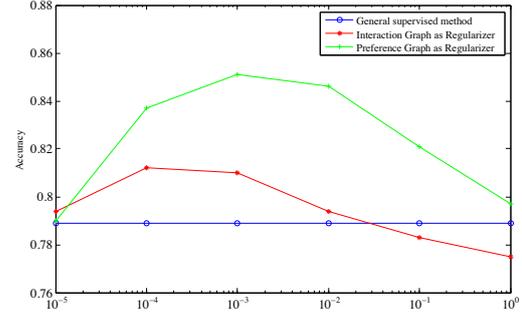


Figure 2: Results with different values of  $\beta$

sitivity.  $\beta$  is the tradeoff weight for graph regularization term. Figure 2 shows the results with different values of  $\beta$ . We can see that when  $\beta$  ranges from  $10^{-4} \sim 10^{-2}$ , the deceptive answer prediction can achieve best results.

## 6 Conclusions and Future Work

In this paper, we discuss the deceptive answer prediction task in Community QA sites. With the manually labeled data set, we first predict the deceptive answers with traditional classification method. Two types of features, including textual features and contextual features, are extracted and analyzed. We also introduce a new user preference graph, constructed based on the user evaluations towards the target answer, such as “helpful” voting and “best answer” selection. A graph regularization method is proposed to incorporate the user preference graph for deceptive answer prediction. The experiments are conducted to discuss the effects of different features. The experiment results also show that the method with user preference graph can achieve more accurate results for deceptive answer prediction.

In the future work, it is interesting to incorporate more features into deceptive answer prediction. It is also important to predict the deceptive question threads, which are posted and answered both by malicious users for product promotion. Malicious user group detection is also an important task in the future.

## References

- Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. 2008. Knowledge sharing and yahoo answers: everyone knows something. In *Proceedings of the 17th international conference on World Wide Web, WWW '08*, pages 665–674, New York, NY, USA. ACM.
- Jiang Bian, Yandong Liu, Ding Zhou, Eugene Agichtein, and Hongyuan Zha. 2009. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *Proceedings of the 18th international conference on World wide web, WWW '09*, pages 51–60, NY, USA. ACM.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.
- Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Comput. Linguist.*, 16:79–85, June.
- S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.
- A. Figueroa and J. Atkinson. 2011. Maximum entropy context models for ranking biographical answers to open-domain definition questions. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.
- F. Maxwell Harper, Daphne Raban, Sheizaf Rafaeli, and Joseph A. Konstan. 2008. Predictors of answer quality in online q&a sites. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, CHI '08*, pages 865–874, New York, NY, USA. ACM.
- Daisuke Ishikawa, Tetsuya Sakai, and Noriko Kando. 2010. *Overview of the NTCIR-8 Community QA Pilot Task (Part I): The Test Collection and the Task*, pages 421–432. Number Part I.
- Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM CIKM conference*, 05, pages 84–90, NY, USA. ACM.
- J. Jeon, W.B. Croft, J.H. Lee, and S. Park. 2006. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 228–235. ACM.
- P. Jurczyk and E. Agichtein. 2007. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM CIKM conference*, pages 919–922. ACM.
- H. Kim, P. Howland, and H. Park. 2006. Dimension reduction in text classification with support vector machines. *Journal of Machine Learning Research*, 6(1):37.
- Fangtao Li, Minlie Huang, Yi Yang, and Xiaoyan Zhu. 2011. Learning to identify review spam. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2488–2493. AAAI Press.
- Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1, COLING '08*, pages 497–504, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Jing Liu, Young-In Song, and Chin-Yew Lin. 2011. Competition-based user expertise score estimation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 425–434. ACM.
- Yue Lu, Panayiotis Tsaparas, Alexandros Ntoulas, and Livia Polanyi. 2010. Exploiting social context for review quality prediction. In *Proceedings of the 19th international conference on World wide web*, pages 691–700. ACM.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29:19–51, March.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November. SIDL-WP-1999-0120.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, pages 311–318, Stroudsburg, PA, USA. ACL.
- Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.
- Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '10*, pages 411–418, New York, NY, USA. ACM.
- X. Si, Z. Gyongyi, and E. Y. Chang. 2010a. Scalable mining of topic-dependent user reputation for improving user generated content search quality. In *Google Technical Report*.

- Xiance Si, Edward Y. Chang, Zoltán Gyöngyi, and Maosong Sun. 2010b. Confucius and its intelligent disciples: integrating social with search. *Proc. VLDB Endow.*, 3:1505–1516, September.
- Young-In Song, Chin-Yew Lin, Yunbo Cao, and Hae-Chang Rim. 2008. Question utility: a novel static ranking of question search. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, pages 1231–1236. AAAI Press.
- Y.I. Song, J. Liu, T. Sakai, X.J. Wang, G. Feng, Y. Cao, H. Suzuki, and C.Y. Lin. 2010. Microsoft research asia with redmond at the ntcir-8 community qa pilot task. In *Proceedings of NTCIR*.
- Wei Wei, Gao Cong, Xiaoli Li, See-Kiong Ng, and Guohui Li. 2011. Integrating community question and answer archives. In *AAAI*.
- Y. Yang and J.O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 412–420. MORGAN KAUFMANN PUBLISHERS.
- Tong Zhang, Alexandrin Popescul, and Byron Dom. 2006. Linear prediction models with graph regularization for web-page categorization. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 821–826. ACM.

# Why-Question Answering using Intra- and Inter-Sentential Causal Relations

Jong-Hoon Oh\* Kentaro Torisawa† Chikara Hashimoto ‡ Motoki Sano§  
Stijn De Saeger¶ Kiyonori Ohtake||

Information Analysis Laboratory  
Universal Communication Research Institute  
National Institute of Information and Communications Technology (NICT)  
{\*rovellia,†torisawa,‡ch,§msano,¶stijn,||kiyonori.ohtake}@nict.go.jp

## Abstract

In this paper, we explore the utility of *intra-* and *inter-sentential causal relations* between terms or clauses as evidence for answering why-questions. To the best of our knowledge, this is the first work that uses both intra- and inter-sentential causal relations for why-QA. We also propose a method for assessing the appropriateness of causal relations as answers to a given question using the semantic orientation of *excitation* proposed by Hashimoto et al. (2012). By applying these ideas to Japanese why-QA, we improved precision by 4.4% against all the questions in our test set over the current state-of-the-art system for Japanese why-QA. In addition, unlike the state-of-the-art system, our system could achieve very high precision (83.2%) for 25% of all the questions in the test set by restricting its output to the confident answers only.

## 1 Introduction

“Why-question answering” (why-QA) is a task to retrieve answers from a given text archive for a why-question, such as “Why are tsunamis generated?” The answers are usually text fragments consisting of one or more sentences. Although much research exists on this task (Girju, 2003; Higashinaka and Isozaki, 2008; Verberne et al., 2008; Verberne et al., 2011; Oh et al., 2012), its performance remains much lower than that of the state-of-the-art factoid QA systems, such as IBM’s Watson (Ferrucci et al., 2010).

In this work, we propose a quite *straightforward* but *novel* approach for such difficult why-QA task. Consider the sentence **A1** in Table 1, which represents the causal relation between the cause, “the ocean’s water mass ..., waves are gen-

<b>A1</b>	[Tsunamis that can cause large coastal inundation are generated] <sub>effect</sub> <u>because</u> [the ocean’s water mass is displaced and, much like throwing a stone into a pond, waves are generated.] <sub>cause</sub>
<b>A2</b>	[Earthquake causes seismic waves which set up the water in motion with a large force.] <sub>cause</sub> <u>This causes</u> [a tsunami]. <sub>effect</sub>
<b>A3</b>	[Tsunamis] <sub>effect</sub> <u>are caused by</u> [the sudden displacement of huge volumes of water.] <sub>cause</sub>
<b>A4</b>	[Tsunamis weaken as they pass through forests] <sub>effect</sub> <u>because</u> [the hydraulic resistance of the trees diminish their energy.] <sub>cause</sub>
<b>A5</b>	[Automakers in Japan suspended production for an array of vehicles] <sub>effect</sub> <u>because</u> [the magnitude 9 earthquake and tsunami hit their country on Friday, March 11, 2011.] <sub>cause</sub>

Table 1: Examples of intra/inter-sentential causal relations. Cause and effect parts of each causal relation, marked with  $[..]_{cause}$  and  $[..]_{effect}$ , are connected by the underlined cue phrases for causality, such as *because*, *this causes*, and *are caused by*.

erated,” and its effect, “Tsunamis ... are generated.” This is a good answer to the question, “Why are tsunamis generated?”, since the effect part is more or less equivalent to the (propositional) content of the question. Our method finds text fragments that include such causal relations with an effect part that resembles a given question and provides them as answers.

Since this idea looks quite intuitive, many people would probably consider it as a solution to why-QA. However, to our surprise, we could not find any previous work on why-QA that took this approach. Some methods utilized the causal relations between terms as evidence for finding answers (i.e., matching a cause term with an answer text and its effect term with a question) (Girju, 2003; Higashinaka and Isozaki, 2008). Other approaches utilized such clue terms for causality as “because” as evidence for finding answers (Murata et al., 2007). However, these algorithms did not check whether an answer candidate, i.e., a text fragment that may be provided as an answer, explicitly contains a complex causal relation sen-

tence with the effect part that resembles a question. For example, **A5** in Table 1 is an incorrect answer to “Why are tsunamis generated?”, but these previous approaches would probably choose it as a proper answer due to “because” and “earthquake” (i.e., a cause of tsunamis). At least in our experimental setting, our approach outperformed these simpler causality-based QA systems.

Perhaps this approach was previously deemed infeasible due to two non-trivial technical challenges. The first challenge is to accurately identify a wide range of causal relations like those in Table 1 in answer candidates. To meet this challenge, we developed a sequence labeling method that identifies not only *intra-sentential causal relations*, i.e., the causal relations between two terms/phrases/clauses expressed in a single sentence (e.g., **A1** in Table 1), but also the *inter-sentential causal relations*, which are the causal relations between two terms/phrases/clauses expressed in two adjacent sentences (e.g., **A2**) in a given text fragment.

The second challenge is assessing the appropriateness of each identified causal relation as an answer to a given question. This is important since the causal relations identified in the answer candidates may have nothing to do with a given question. In this case, we have to reject these causal relations because they are inappropriate as an answer to the question. When a single answer candidate contains many causal relations, we also have to select the appropriate ones. Consider the causal relations in **A1–A4**. Those in **A1–A3** are appropriate answers to “Why are tsunamis generated?”, but not the one in **A4**. To assess the appropriateness, the system must recognize *textual entailment*, i.e., “tsunamis (are) generated” in the question is entailed by all “tsunamis are generated” in **A1**, “cause a tsunami” in **A2** and “tsunamis are caused” in **A3** but not by “tsunamis weaken” in **A4**. This quite difficult task is currently being studied by many researchers in the RTE field (Androutsopoulos and Malakasiotis, 2010; Dagan et al., 2010; Shima et al., 2011; Bentivogli et al., 2011). To meet this challenge, we developed a relatively simple method that can be seen as a lightweight approximation for this difficult RTE task, using excitation polarities (Hashimoto et al., 2012).

Through our experiments on Japanese why-QA, we show that a combination of the above methods

can improve why-QA accuracy. In addition, our proposed method can be successfully combined with other approaches to why-QA and can contribute to higher accuracy. As a final result, we improved the precision by 4.4% against all the questions in our test set over the current state-of-the-art system of Japanese why-QA (Oh et al., 2012). The difference in the performance became much larger when we only compared the highly confident answers of each system. When we made our system provide only its confident answers according to their confidence score given by our system, the precision of these confident answers was 83.2% for 25% of all the questions in our test set. In the same setting, the precision of the state-of-the-art system (Oh et al., 2012) was only 62.4%.

## 2 Related Work

Although there were many previous works on the acquisition of intra- and inter-sentential causal relations from texts (Khoo et al., 2000; Girju, 2003; Inui and Okumura, 2005; Chang and Choi, 2006; Torisawa, 2006; Blanco et al., 2008; De Saeger et al., 2009; De Saeger et al., 2011; Riaz and Girju, 2010; Do et al., 2011; Radinsky et al., 2012), their application to why-QA was limited to causal relations between terms (Girju, 2003; Higashinaka and Isozaki, 2008).

As previous attempts to improve why-QA performance, such semantic knowledge as WordNet synsets (Verberne et al., 2011), semantic word classes (Oh et al., 2012), sentiment analysis (Oh et al., 2012), and causal relations between terms (Girju, 2003; Higashinaka and Isozaki, 2008) has been used. These previous studies took basically bag-of-words approaches and used the semantic knowledge to identify certain semantic associations using terms and n-grams. On the other hand, our method explicitly identifies intra- and inter-sentential causal relations between terms/phrases/clauses that have complex structures and uses the identified relations to answer a why-question. In other words, our method considers more complex linguistic structures than those used in the previous studies. Note that our method can complement the previous approaches. Through our experiments, we showed that it is possible to achieve a higher precision by combining our proposed method with bag-of-words approaches considering semantic word classes and sentiment analysis in our previous work (Oh et al.,

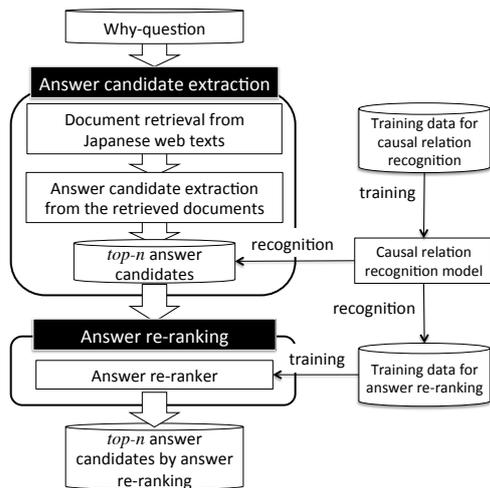


Figure 1: System architecture

2012).

### 3 System Architecture

We first describe the system architecture of our QA system before describing our proposed method. It is composed of two components: answer candidate extraction and answer re-ranking (Fig. 1). This architecture is basically the same as that used in our previous work (Oh et al., 2012). We extended our previous work by introducing causal relations recognized from answer candidates to the answer re-ranking. The features used in our previous work are very different from those in this work, and we found that combining both improves accuracy.

**Answer candidate extraction:** In our previous work, we implemented the method of Murata et al. (2007) for our answer candidate extractor. We retrieved documents from Japanese web texts using *Boolean AND* and *OR* queries generated from the content words in why-questions. Then we extracted passages of five sentences from these retrieved documents and ranked them with the ranking function proposed by Murata et al. (2007). This method ranks a passage higher when it contains more query terms that are closer to each other in the passage. We used a set of clue terms, including the Japanese counterparts of *cause* and *reason*, as query terms for the ranking. The top ranked passages are regarded as *answer candidates* in the answer re-ranking. See Murata et al. (2007) for more details.

**Answer re-ranking:** Re-ranking the answer candidates is done by a supervised classifier (SVMs) (Vapnik, 1995). In our previous work, we

employed three types of features for training the re-ranker: morphosyntactic features ( $n$ -grams of morphemes and syntactic dependency chains), semantic word class features (semantic word classes obtained by automatic word clustering (Kazama and Torisawa, 2008)) and sentiment polarity features (word and phrase polarities). Here, we used semantic word classes and sentiment polarities for identifying such semantic associations between a why-question and its answer as “if a disease’s name appears in a question, then answers that include nutrient names are more likely to be correct” by semantic word classes and “if something undesirable happens, the reason is often also something undesirable” by sentiment polarities. In this work, we propose causal relation features generated from intra- and inter-sentential causal relations in answer candidates and use them along with the features proposed in our previous work for training our re-ranker.

### 4 Causal Relations for Why-QA

We describe causal relation recognition in Section 4.1 and describe the features (of our re-ranker) generated from causal relations in Section 4.2.

#### 4.1 Causal Relation Recognition

We restrict causal relations to those expressed by such cue phrases for causality as (the Japanese counterparts of) *because* and *as a result* like in the previous work (Khoo et al., 2000; Inui and Okumura, 2005) and recognize them in the following two steps: extracting causal relation candidates and recognizing causal relations from these candidates.

##### 4.1.1 Extracting Causal Relation Candidates

We identify cue phrases for causality in answer candidates using the regular expressions in Table 2. Then, for each identified cue phrase, we extract three sentences as a causal relation candidate, where one contains the cue phrase and the other two are the previous and next sentences in the answer candidates. When there is more than one cue phrase in an answer candidate, we use all of them for extracting the causal relation candidates, assuming that each of the cue phrases is linked to different causal relations. We call a cue phrase used for extracting a causal relation candidate a *c-marker* (*causality marker*) of the candidate to distinguish it from the other cue phrases in the same causal relation candidate.

Regular expressions	Examples
(D の)? ため P?	ため (for), のため (for), そのため (as a result), のために (for)
ので	ので (since or because of)
こと (から で)	ことから (from the fact that), ことで (by the fact that)
(から ため) C	からだ (because), ためた (It is because)
D? RCT (P C)+	理由は (the reason is), 原因だ (is the cause), この理由から (from this reason)

Table 2: Regular expressions for identifying cue phrases for causality. **D**, **P** and **C** represent demonstratives (e.g., この (this) and その (that)), postpositions (including case markers such as が (nominative), の (genitive)), and copula (e.g., です (is) and である (is)) in Japanese, respectively. **RCT**, which represents Japanese terms meaning *reason*, *cause*, or *thanks to*, is defined as follows: **RCT** = {理由 (reason), 原因 (cause), 要因 (cause), 引き金 (cause), おかげ (thanks to), せい (thanks to), わけ (reason)}.

#### 4.1.2 Recognizing Causal Relations

Next, we recognize the spans of the cause and effect parts of a causal relation linked to a c-marker. We regard this task as a sequence labeling problem and use Conditional Random Fields (CRFs) (Lafferty et al., 2001) as a machine learning framework. In our task, CRFs take three sentences of a causal relation candidate as input and generate their cause-effect annotations with a set of possible cause-effect IOB labels, including Begin-Cause (B-C), Inside-Cause (I-C), Begin-Effect (B-E), Inside-Effect (I-E), and Outside (O). Fig 2 shows an example of such sequence labeling. Although this example is about sequential labeling shown on English sentences for ease of explanation, it was actually done on Japanese sentences.

We used the three types of feature sets in Table 3 for training the CRFs, where  $j$  is in the range of  $i - 4 \leq j \leq i + 4$  for current position  $i$  in a causal relation candidate.

Type	Features
Morphological feature	$m_j, m_j^{j+1}, pos_j, pos_j^{j+1}$
Syntactic feature	$s_j, s_j^{j+1}, b_j, b_j^{j+1}$
C-marker feature	$(m_j, cm), (m_j^{j+1}, cm)$ $(s_j, cm), (s_j^{j+1}, cm)$

Table 3: Features for training CRFs, where  $x_j^{j+1} = x_j x_{j+1}$

**Morphological features:**  $m_j$  and  $pos_j$  in Table 3 represent the  $j^{th}$  morpheme and the POS tag.

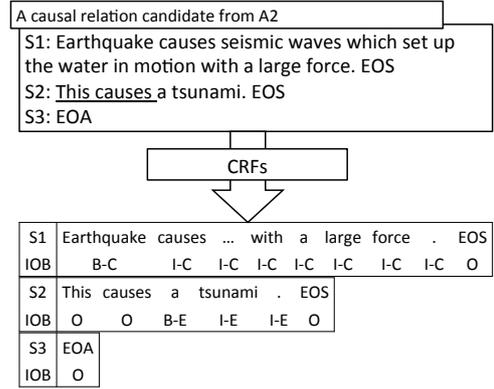


Figure 2: Recognizing causal relations by sequence labeling: Underlined text *This causes* represents a c-marker, and EOS and EOA represent *end-of-sentence* and *end-of-answer candidates*.

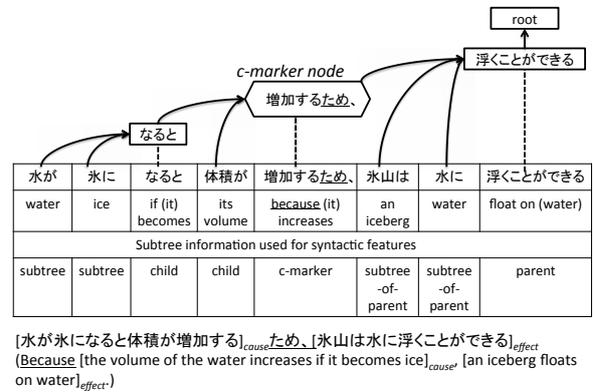


Figure 3: Example of syntactic information related to a c-marker used for syntactic features

We use JUMAN<sup>1</sup>, a Japanese morphological analyzer, for generating our morphological features.

**Syntactic features:** The span of the causal relations in a given causal relation candidate strongly depends on the c-marker in the candidate. Especially for intra-sentential causal relations, their cause and effect parts often appear in the subtrees of the c-marker's node or those of the c-marker's parent node in a syntactic dependency tree structure. Fig. 3 shows an example that follows this observation, where the c-marker node is represented in a hexagon and the other nodes are in a rectangle. Note that each node in Fig. 3 is a word phrase (called a *bunsetsu*), which is the smallest unit of syntactic analysis in Japanese. A *bunsetsu* is a syntactic constituent composed of a content word and several function words such as postpositions and case markers. Syntactic dependency is represented by an arrow in Fig. 3. For example, there is syntactic dependency from word phrase 水が

<sup>1</sup> <http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN>

(water) になると (if (it) becomes), i.e., 水が<sup>dep</sup> になると. We encode this subtree information into  $s_j$ , which is the syntactic information of a word phrase to which the  $j^{\text{th}}$  morpheme belongs.  $s_j$  only has one of six values: 1) the c-marker’s node (*c-marker*), 2) the c-marker’s child node (*child*), 3) the c-marker’s parent node (*parent*), 4) in the c-marker’s subtree but not the c-marker’s child node (*subtree*), 5) in the subtree of the c-marker’s parent node but not the c-marker’s node (*subtree-of-parent*) and 6) the others (*others*).  $b_j$  is the word phrase information of the  $j^{\text{th}}$  morpheme ( $m_j$ ) that represents whether  $m_j$  is in the beginning or inside a word phrase. For generating our syntactic features, we use KNP<sup>2</sup>, a Japanese syntactic dependency parser.

**C-marker features:** As our c-marker features, we use a pair composed of c-marker  $cm$  and one of the following:  $m_j$ ,  $m_j^{j+1}$ ,  $s_j$ , or  $s_j^{j+1}$ .

## 4.2 Causal Relation Features

We use terms, partial trees (in a syntactic dependency tree structure), and the semantic orientation of excitation (Hashimoto et al., 2012) to assess the appropriateness of each causal relation obtained by our causal relation recognizer as an answer to a given question. Finding answers with term matching and partial tree matching has been used in the literature of question answering (Girju, 2003; Narayanan and Harabagiu, 2004; Moschitti et al., 2007; Higashinaka and Isozaki, 2008; Verberne et al., 2008; Surdeanu et al., 2011; Verberne et al., 2011; Oh et al., 2012), while that with the excitation polarity is proposed in this work.

We use three types of features. Each feature type expresses the causal relations in an answer candidate that are determined to be appropriate as answers to a given question by term matching ( $tf_1$ – $tf_4$ ), partial tree matching ( $pf_1$ – $pf_4$ ) and excitation polarity matching ( $ef_1$ – $ef_4$ ). We call these causal relations used for generating our causal relation features *candidates of an appropriate causal relation* in this section. Note that if one answer candidate has more than one candidate of an appropriate causal relation found by one matching method, we generated features for each appropriate candidate and merged all of them for the answer candidate.

Type	Description
$tf_1$	word $n$ -grams of causal relations
$tf_2$	word class version of $tf_1$
$tf_3$	indicator for the existence of candidates of an appropriate causal relation identified by term matching in an answer candidate
$tf_4$	number of matched terms in candidates of an appropriate causal relation
$pf_1$	syntactic dependency $n$ -grams ( $n$ dependency chain) of causal relations
$pf_2$	word class version of $pf_1$
$pf_3$	indicator for the existence of candidates of an appropriate causal relation identified by partial tree matching in an answer candidate
$pf_4$	number of matched partial trees in candidates of an appropriate causal relation
$ef_1$	types of noun-polarity pairs shared by causal relations and the question
$ef_2$	$ef_1$ coupled with each noun’s word class
$ef_3$	indicator for the existence of candidates of an appropriate causal relation identified by excitation polarity matching in an answer candidate
$ef_4$	number of noun-polarity pairs shared by the question and the candidates of an appropriate causal relation

Table 4: Causal relation features:  $n$  in  $n$ -grams is  $n = \{2, 3\}$  and  $n$ -grams in an effect part are distinguished from those in a cause part.

### 4.2.1 Term Matching

Our term matching method judges that a causal relation is a candidate of an appropriate causal relation if its effect part contains at least one content word (nouns, verbs, and adjectives) in the question. For example, all the causal relations of **A1**–**A4** in Table 1 are candidates of an appropriate causal relation to the question, “Why is a tsunami generated?”, by term matching with question term *tsunami*.

$tf_1$ – $tf_4$  are generated from candidates of an appropriate causal relation identified by term matching. The  $n$ -grams of  $tf_1$  and  $tf_2$  are restricted to those containing at least one content word in a question. We distinguish this matched word from the other words by replacing it with QW, a special symbol representing a word in the question. For example, word 3-gram “this/cause/QW” is extracted from *This causes tsunamis* in **A2** for “Why is a tsunami generated?” Further, we create a word class version of word  $n$ -grams by converting the words in these word  $n$ -grams into their corresponding word class using the semantic word classes (500 classes for 5.5 million nouns) from our previous work (Oh et al., 2012). These word classes were created by applying the automatic word clustering method of Kazama and Torisawa (2008) to 600 million Japanese web pages. For example, the word class version of word 3-gram

<sup>2</sup> <http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP>

“this/cause/QW” is “this/cause/QW,WC<sub>tsunami</sub>”, where WC<sub>tsunami</sub> represents the word class of a tsunami.  $tf_3$  is a binary feature that indicates the existence of candidates of an appropriate causal relation identified by term matching in an answer candidate.  $tf_4$  represents the degree of the relevance of the candidates of an appropriate causal relation measured by the number of matched terms: one, two, and more than two.

### 4.2.2 Partial Tree Matching

Our partial tree matching method judges a causal relation as a candidate of an appropriate causal relation if its effect part contains at least one partial tree in a question, where the partial tree covers more than one content word. For example, only the causal relation **A1** among **A1–A4** is a candidate of an appropriate causal relation for question “Why are tsunamis generated?” by partial tree matching because only its effect part contains partial tree “tsunamis  $\xrightarrow{dep}$  (are) generated” of the question.

$pf_1$ – $pf_4$  are generated from candidates of an appropriate causal relation identified by the partial tree matching. The syntactic dependency  $n$ -grams in  $pf_1$  and  $pf_2$  are restricted to those that contain at least one content word in a question. We distinguish this matched content word from the other content words in the  $n$ -gram by converting it to QW, which represents a content word in the question. For example, syntactic dependency 2-gram “QW  $\xrightarrow{dep}$  cause” and its word class version “QW,WC<sub>tsunami</sub>  $\xrightarrow{dep}$  cause” are extracted from *Tsunamis that can cause* in **A1**.  $pf_3$  is a binary feature that indicates whether an answer candidate contains candidates of an appropriate causal relation identified by partial tree matching.  $pf_4$  represents the degree of the relevance of the candidate of an appropriate causal relation measured by the number of matched partial trees: one, two, and more than two.

### 4.2.3 Excitation Polarity Matching

Hashimoto et al. (2012) proposed a semantic orientation called *excitation polarities*. It classifies predicates with their argument position (called *templates*) into *excitatory*, *inhibitory* and *neutral*. In the following, we denote a template as “[*argument position, predicate*].” According to Hashimoto’s definition, excitatory templates imply that the function, effect, purpose, or the role of

an entity filling an argument position in the templates is activated/enhanced. On the contrary, inhibitory templates imply that the effect, purpose or the role of an entity is deactivated/suppressed. Neutral templates are those that neither activate nor suppress the function of an argument.

We assume that the *meanings* of a text can be *roughly* captured by checking whether each noun in the text is *activated* or *suppressed* in the sense of the excitation polarity framework, where the activation and suppression of each entity (or noun) can be detected by looking at the excitation polarities of the templates that are filled by the entity. For instance, effect part “tsunamis that can cause large coastal inundation are generated” of **A1** roughly means that “tsunamis” are *activated* and “inundation” is (or can be) *activated*. This activation/suppression configuration of the nouns is *consistent* with sentence “tsunamis are caused” in which “tsunamis” are *activated*. This consistency suggests that **A1** is a good answer to question “Why are tsunamis caused?”, although the “tsunamis” are modified by different predicates; “cause” and “generate.” On the other hand, effect part “tsunamis weaken as they pass through forests” of **A4** implies that “tsunamis” are *suppressed*. This suggests that **A4** is not a good answer to “Why are tsunamis caused?” Note that the consistency checking between activation/suppression configurations of nouns<sup>3</sup> in texts can be seen as a rough but lightweight approximation of the recognition of textual entailments or paraphrases.

Following the definition of excitation polarity in Hashimoto et al. (2012), we manually classified templates<sup>4</sup> to each polarity type and obtained 8,464 excitatory templates, such as [が, 増える] ([subject, increase]) and [が, 向上する] ([subject, improve]), 2,262 inhibitory templates, such as [を, 防ぐ] ([object, prevent]) and [が, 死ぬ] ([subject, die]), and 7,230 neutral templates such as [を, 考える] ([object, consider]). With these templates, we obtain activation/suppression configurations (including neutral) for the nouns in the causal relations in the answer candidates and ques-

<sup>3</sup> Because the activation/suppression configurations of nouns come from an excitation polarity of templates, “[*argument position, predicate*],” the semantics of verbs in the templates are implicitly considered in this consistency checking.

<sup>4</sup> Varga et al. (2013) has used the same templates as ours, except they restricted their excitation/inhibitory templates to those whose polarity is consistent with that given by the automatic acquisition method of Hashimoto et al. (2012).

tions.

Next, we assume that a causal relation is appropriate as an answer to a question if the effect part of the causal relation and the question share at least one common noun with the same polarity. More detailed information concerning the configurations of all the nouns in all the candidates of an *appropriate* causal relation (including their cause parts) and the question are encoded into our feature set  $ef_1$ – $ef_4$  in Table 4 and the final judgment is done by our re-ranker.

For generating  $ef_1$  and  $ef_2$ , we classified all the nouns coupled with activation/suppression/neutral polarities in a causal relation into three types: SAME (the question contains the same noun with the same polarity), DiffPOL (the question contains the same noun with different polarity), and OTHER (the others).  $ef_1$  indicates whether each type of noun-polarity pair exists in a causal relation. Note that the types for the effect and cause parts are represented in distinct features.  $ef_2$  is the same as  $ef_1$  except that the types are augmented with the word classes of the corresponding nouns. In other words,  $ef_2$  indicates whether each type of noun-polarity pair exists in the causal relation for each word class.  $ef_3$  indicates the existence of candidates of an appropriate causal relation identified by this matching scheme, and  $ef_4$  represents the number of noun-polarity pairs shared by the question and the candidates of an appropriate causal relations (one, two, and more than two).

## 5 Experiments

We experimented with causal relation recognition and why-QA with our causal relation features.

### 5.1 Data Set for Why-Question Answering

For our experiments, we used the same why-QA data set as the one used in our previous work (Oh et al., 2012). This why-QA data set is composed of 850 Japanese why-questions and their top-20 answer candidates obtained by answer candidate extraction from 600 million Japanese web pages. Three annotators checked the top-20 answer candidates of these 850 questions and the final judgment was made by their majority vote. Their inter-rater agreement by Fleiss’ kappa reported in Oh et al. (2012) was substantial ( $\kappa = 0.634$ ). Among the 850 questions, 250 why-questions were extracted from the Japanese version of *Yahoo! Answers*, and another 250 were created by annotators. In

our previous work, we evaluated the system with these 500 questions and their answer candidates as training and test data in 10-fold cross-validation. The other 350 why-questions were manually built from passages describing the causes or reasons of events/phenomena. These questions and their answer candidates were used as additional training data for testing subsamples in each fold during the 10-fold cross-validation. In our why-QA experiments, we evaluated our why-QA system with the same settings.

### 5.2 Data Set for Causal Relation Recognition

We built a data set composed of manually annotated causal relations for evaluating our causal relation recognition. As source data for this data set, we used the same 10-fold data that we used for evaluating our why-QA (500 questions and their answer candidates). We extracted the causal relation candidates from the answer candidates in each fold, and then our annotator (not an author) manually marked the span of the cause and effect parts of a causal relation for each causal relation candidate, keeping in mind that the causal relation must be expressed in terms of a c-marker in a given causal relation candidate. Finally, we had a data set made of 16,051 causal relation candidates, 8,117 of which had a true causal relation; the number of intra- and inter-sentential causal relations were 7,120 and 997, respectively.

Note that this data set can be partitioned into ten folds by using the 10-fold partition of its source data. We performed 10-fold cross validation to evaluate our causal relation recognition with this 10-fold data.

### 5.3 Causal Relation Recognition

We used CRF++<sup>5</sup> for training our causal relation recognizer. In our evaluation, we judged a system’s output as correct if both spans of the cause and effect parts overlapped those in the gold standard. Evaluation was done by precision, recall, and  $F_1$ .

	Precision	Recall	$F_1$
BASELINE	41.9	61.0	49.7
INTRA-SENT	84.5	75.4	79.7
INTER-SENT	80.2	52.6	63.6
ALL	83.8	71.1	77.0

Table 5: Results of causal relation recognition (%)

Table 5 shows the result. BASELINE represents

<sup>5</sup> <http://code.google.com/p/crfpp/>

the result for our baseline system that recognizes a causal relation by simply taking the two phrases adjacent to a c-marker (i.e., before and after) as cause and effect parts of the causal relation. We assumed that the system had an oracle for judging correctly whether each phrase is a cause part or an effect part. In other words, we judged that a causal relation recognized by BASELINE is correct if both cause and effect parts in the gold standard are adjacent to a c-marker. INTRA-SENT and INTER-SENT represent the results for intra- and inter-sentential causal relations and ALL represents the result for the both causal relations by our method. From these results, we confirmed that our method recognized both intra- and inter-sentential causal relations with over 80% precision, and it significantly outperformed our baseline system in both precision and recall rates.

	Precision	Recall	$F_1$
ALL-“MORPH”	80.8	66.4	72.9
ALL-“SYNTACTIC”	82.9	67.0	74.1
ALL-“C-MARKER”	76.3	51.4	61.4
ALL	<b>83.8</b>	<b>71.1</b>	<b>77.0</b>

Table 6: Ablation test results for causal relation recognition (%)

We also investigated the contribution of the three types of features used in our causal relation recognition to the performance. We evaluated the performance when we removed one of the three types of features (ALL-“MORPH”, ALL-“SYNTACTIC” and ALL-“C-MARKER”) and compared the results in these settings with the one when all the feature sets were used (ALL). Table 6 shows the result. We confirmed that all the feature sets improved the performance, and we got the best performance when using all of them. We used the causal relations obtained from the 10-fold cross validation for our why-QA experiments.

#### 5.4 Why-Question Answering

We performed why-QA experiments to confirm the effectiveness of intra- and inter-sentential causal relations in a why-QA task. In this experiment, we compared five systems: four baseline systems (MURATA, OURCF, OH and OH+PREVCF) and our proposed method (PROPOSED).

MURATA corresponds to our answer candidate extraction.

OURCF uses a re-ranker trained with only our

causal relation features.

OH, which represents our previous work (Oh et al., 2012), has a re-ranker trained with morphosyntactic, semantic word class, and sentiment polarity features.

OH+PREVCF is a system with a re-ranker trained with the features used in OH and with the causal relation feature proposed in Higashinaka and Isozaki (2008). The causal relation feature includes an indicator that determines whether the causal relations between two terms appear in a question-answer pair; cause in an answer and its effect in a question. We acquired the causal relation instances (between terms) from 600 million Japanese web pages using the method of De Saeger et al. (2009) and exploited the *top-100,000* causal relation instances in this system.

PROPOSED has a re-ranker trained with our causal relation features as well as the three types of features proposed in Oh et al. (2012). Comparison between OH and PROPOSED reveals the contribution of our causal relation features to why-QA.

We used TinySVM<sup>6</sup> with a linear kernel for training the re-rankers in OURCF, OH, OH+PREVCF and PROPOSED. Evaluation was done by P@1 (Precision of the top-answer) and Mean Average Precision (MAP); they are the same measures used in Oh et al. (2012). P@1 measures how many questions have a correct top-answer candidate. MAP measures the overall quality of the top-20 answer candidates. As mentioned in Section 5.1, we used 10-fold cross-validation with the same setting as the one used in Oh et al. (2012) for our experiments.

	P@1	MAP
MURATA	22.2	27.0
OURCF	27.8	31.4
OH	37.4	39.1
OH+PREVCF	37.4	38.9
PROPOSED	<b>41.8</b>	<b>41.0</b>

Table 7: Why-QA results (%)

Table 7 shows the evaluation results. Our proposed method outperformed the other four systems and improved P@1 by 4.4% over OH, which is the-state-of-the-art system for Japanese why-

<sup>6</sup> <http://chasen.org/~taku/software/TinySVM/>

QA. OURCF showed the performance improvement over MURATA. Although this suggests the effectiveness of our causal relation features, the overall performance of OURCF was lower than that of OH. OH+PREVCF outperformed neither OH nor PROPOSED. This suggests that our approach is more effective than previous causality-based approaches (Girju, 2003; Higashinaka and Isozaki, 2008), at least in our setting.

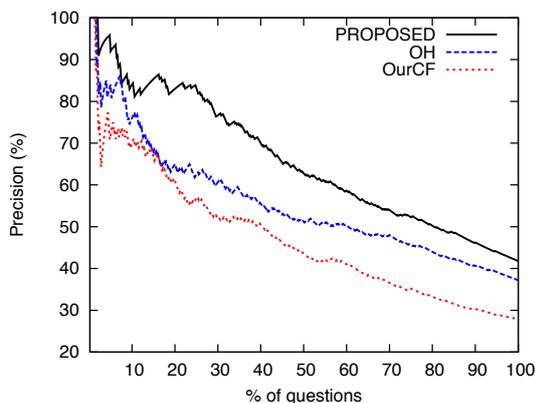


Figure 4: Effect of causal relation features on the top-answers

We also compared *confident* answers of OURCF, OH, and PROPOSED by making each system provide only the  $k$  confident top-answers (for  $k$  questions) selected by their SVM scores given by each system’s re-ranker. This reduces the number of questions that can be answered by a system, but the top-answers become more reliable as  $k$  decreases. Fig. 4 shows this result, where the  $x$  axis represents the percentage of questions (against all the questions in our test set) whose top-answers are given by each system, and the  $y$  axis represents the precision of the top-answers at a certain point on the  $x$  axis. When both systems provided top-answers for 25% of all the questions in our test set, our method achieved 83.2% precision, which is much higher than OH’s (62.4%). This experiment confirmed that our causal relation features were also effective in improving the quality of the highly confident answers.

However, the high precision by our method was bound to confident answers for a small number of questions, and the difference in the precision between OH and PROPOSED in Fig. 4 became smaller as we considered more answers with lower confidence. We think that one of the reasons is the relatively small coverage of the excitation polarity lexicon, a core resource in our excitation polarity

matching. We are planning to enlarge the lexicon to deal with this problem.

Next, we investigated the contribution of the intra- and inter-sentential causal relations to the performance of our method. We used only one of the two types of causal relations for generating causal relation features (INTRA-SENT and INTER-SENT) for training our re-ranker and compared the results in these settings with the one when both were used (ALL (PROPOSED)). Table 8 shows the result. Both intra- and inter-sentential causal relations contributed to the performance improvement.

	P@1	MAP
INTER-SENT	39.0	39.7
INTRA-SENT	40.4	40.5
ALL (PROPOSED)	<b>41.8</b>	<b>41.0</b>

Table 8: Results with/without intra- and inter-sentential causal relations (%)

We also investigated the contributions of the three types of causal relation features by ablation tests (Table 9). When we do not use the features by excitation polarity matching (ALL- $\{ef_1-ef_4\}$ ), the performance is the worst. This implies that the contribution of excitation polarity matching exceeds the other two.

	P@1	MAP
ALL- $\{tf_1-tf_4\}$	40.8	40.7
ALL- $\{pf_1-pf_4\}$	41.0	40.9
ALL- $\{ef_1-ef_4\}$	39.6	40.5
ALL (PROPOSED)	<b>41.8</b>	<b>41.0</b>

Table 9: Ablation test results for why-QA (%)

## 6 Conclusion

In this paper, we explored the utility of intra- and inter-sentential causal relations for ranking answer candidates to why-questions. We also proposed a method for assessing the appropriateness of causal relations as answers to a given question using the semantic orientation of excitation. Through experiments, we confirmed that these ideas are effective for improving why-QA, and our proposed method achieved 41.8% P@1, which is 4.4% improvement over the current state-of-the-art system of Japanese why-QA. We also showed that our system achieved 83.2% precision for its confident answers, when it only provided its confident answers for 25% of all the questions in our test set.

## References

- Ion Androutsopoulos and Prodrornos Malakasiotis. 2010. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research (JAIR)*, 38(1):135–187.
- Luisa Bentivogli, Peter Clark, Ido Dagan, Hoa Dang, and Danilo Giampiccolo. 2011. The seventh pascal recognizing textual entailment challenge. In *Proceedings of TAC*.
- E. Blanco, N. Castell, and Dan I. Moldovan. 2008. Causal relation extraction. In *Proceedings of LREC'08*.
- Du-Seong Chang and Key-Sun Choi. 2006. Incremental cue phrase learning and bootstrapping method for causality extraction using cue phrase and word pair probabilities. *Information Processing and Management*, 42(3):662–678.
- Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 16(1):1–17.
- Stijn De Saeger, Kentaro Torisawa, Jun'ichi Kazama, Kow Kuroda, and Masaki Murata. 2009. Large scale relation acquisition using class dependent patterns. In *Proceedings of ICDM '09*, pages 764–769.
- Stijn De Saeger, Kentaro Torisawa, Masaaki Tsuchida, Jun'ichi Kazama, Chikara Hashimoto, Ichiro Yamada, Jong Hoon Oh, István Varga, and Yulan Yan. 2011. Relation acquisition using word classes and partial patterns. In *Proceedings of EMNLP '11*, pages 825–835.
- Quang Xuan Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of EMNLP '11*, pages 294–303.
- David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building Watson: An overview of the DeepQA project. *AI Magazine*, 31(3):59–79.
- Roxana Girju. 2003. Automatic detection of causal relations for question answering. In *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering*, pages 76–83.
- Chikara Hashimoto, Kentaro Torisawa, Stijn De Saeger, Jong-Hoon Oh, and Jun'ichi Kazama. 2012. Excitatory or inhibitory: A new semantic orientation extracts contradiction and causality from the web. In *Proceedings of EMNLP-CoNLL '12*.
- Ryuichiro Higashinaka and Hideki Isozaki. 2008. Corpus-based question answering for why-questions. In *Proceedings of IJCNLP '08*, pages 418–425.
- Takashi Inui and Manabu Okumura. 2005. Investigating the characteristics of causal relations in Japanese text. In *In Annual Meeting of the Association for Computational Linguistics (ACL) Workshop on Frontiers in Corpus Annotations II: Pie in the Sky*.
- Jun'ichi Kazama and Kentaro Torisawa. 2008. Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *Proceedings of ACL-08: HLT*, pages 407–415.
- Christopher S. G. Khoo, Syin Chan, and Yun Niu. 2000. Extracting causal knowledge from a medical database using graphical patterns. In *Proceedings of ACL '00*, pages 336–343.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML '01*, pages 282–289.
- Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings of ACL '07*, pages 776–783.
- Masaki Murata, Sachiyo Tsukawaki, Toshiyuki Kanamaru, Qing Ma, and Hitoshi Isahara. 2007. A system for answering non-factoid Japanese questions by using passage retrieval weighted based on type of answer. In *Proceedings of NTCIR-6*.
- Srini Narayanan and Sanda Harabagiu. 2004. Question answering based on semantic structures. In *Proceedings of COLING '04*, pages 693–701.
- Jong-Hoon Oh, Kentaro Torisawa, Chikara Hashimoto, Takuya Kawada, Stijn De Saeger, Jun'ichi Kazama, and Yiyou Wang. 2012. Why question answering using sentiment analysis and word classes. In *Proceedings of EMNLP-CoNLL '12*, pages 368–378.
- Kira Radinsky, Sagie Davidovich, and Shaul Markovitch. 2012. Learning causality for news events prediction. In *Proceedings of WWW '12*, pages 909–918.
- Mehwish Riaz and Roxana Girju. 2010. Another look at causality: Discovering scenario-specific contingency relationships with no supervision. In *ICSC '10*, pages 361–368.
- Hideki Shima, Hiroshi Kanayama, Cheng wei Lee, Chuan jie Lin, Teruko Mitamura, Yusuke Miyao, Shuming Shi, and Koichi Takeda. 2011. Overview of NTCIR-9 RITE: Recognizing Inference in TExt. In *Proceedings of NTCIR-9*.
- Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2):351–383.

- Kentaro Torisawa. 2006. Acquiring inference rules with temporal constraints by using japanese coordinated sentences and noun-verb co-occurrences. In *Proceedings of HLT-NAACL '06*, pages 57–64.
- Vladimir N. Vapnik. 1995. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.
- Istvan Varga, Motoki Sano, Kentaro Torisawa, Chikara Hashimoto, Kiyonori Ohtake, Takao Kawai, Jong-Hoon Oh, and Stijn De Saeger. 2013. Aid is out there: Looking for help from tweets during a large scale disaster. In *Proceedings of ACL '13*.
- Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2008. Using syntactic information for improving why-question answering. In *Proceedings of COLING '08*, pages 953–960.
- Suzan Verberne, Lou Boves, and Wessel Kraaij. 2011. Bringing why-qa to web search. In *Proceedings of ECIR '11*, pages 491–496.

# Question Answering Using Enhanced Lexical Semantic Models

Wen-tau Yih Ming-Wei Chang Christopher Meek Andrzej Pastusiak

Microsoft Research  
Redmond, WA 98052, USA

{scottyih, minchang, meek, andrzejp}@microsoft.com

## Abstract

In this paper, we study the *answer sentence selection* problem for question answering. Unlike previous work, which primarily leverages syntactic analysis through dependency tree matching, we focus on improving the performance using models of lexical semantic resources. Experiments show that our systems can be consistently and significantly improved with rich lexical semantic information, regardless of the choice of learning algorithms. When evaluated on a benchmark dataset, the MAP and MRR scores are increased by 8 to 10 points, compared to one of our baseline systems using only surface-form matching. Moreover, our best system also outperforms previous work that makes use of the dependency tree structure by a wide margin.

## 1 Introduction

Open-domain question answering (QA), which fulfills a user's information need by outputting direct answers to natural language queries, is a challenging but important problem (Etzioni, 2011). State-of-the-art QA systems often implement a complicated pipeline architecture, consisting of question analysis, document or passage retrieval, answer selection and verification (Ferrucci, 2012; Moldovan et al., 2003). In this paper, we focus on one of the key subtasks – *answer sentence selection*. Given a question and a set of candidate sentences, the task is to choose the *correct* sentence that contains the exact answer and can sufficiently support the answer choice. For instance, although both of the following sentences contain the answer “Jack Lemmon” to the question “Who won the best actor Oscar in 1973?” only the first sentence is correct.

**A1:** Jack Lemmon won the Academy Award for Best Actor for Save the Tiger (1973).

**A2:** Oscar winner Kevin Spacey said that Jack Lemmon is remembered as always making time for other people.

One of the benefits of answer sentence selection is that the output can be provided directly to the user. Instead of outputting only the answer, returning the whole sentence often adds more value as the user can easily verify the correctness without reading a lengthy document.

Answer sentence selection can be naturally reduced to a semantic text matching problem. Conceptually, we would like to measure how close the question and sentence can be *matched* semantically. Due to the variety of word choices and inherent ambiguities in natural languages, bag-of-words approaches with simple surface-form word matching tend to produce brittle results with poor prediction accuracy (Bilotti et al., 2007). As a result, researchers put more emphasis on exploiting both the syntactic and semantic structure in questions/sentences. Representative examples include methods based on deeper semantic analysis (Shen and Lapata, 2007; Moldovan et al., 2007) and on tree edit-distance (Punyakank et al., 2004; Heilman and Smith, 2010) and quasi-synchronous grammar (Wang et al., 2007) that match the dependency parse trees of questions and sentences. However, such approaches often require more computational resources. In addition to applying a syntactic or semantic parser during run-time, finding the best matching between structured representations of sentences is not trivial. For example, the computational complexity of tree matching is  $O(V^2L^4)$ , where  $V$  is the number of nodes and  $L$  is the maximum depth (Tai, 1979).

Instead of focusing on the high-level semantic representation, we turn our attention in this work to improving the shallow semantic compo-

ment, *lexical semantics*. We formulate answer selection as a semantic matching problem with a latent word-alignment structure as in (Chang et al., 2010) and conduct a series of experimental studies on leveraging recently proposed lexical semantic models. Our main contributions in this work are two key findings. First, by incorporating the abundant information from a variety of lexical semantic models, the answer selection system can be enhanced substantially, regardless of the choice of learning algorithms and settings. Compared to the previous work, our latent alignment model improves the result on a benchmark dataset by a wide margin – the mean average precision (MAP) and mean reciprocal rank (MRR) scores are increased by 25.6% and 18.8%, respectively. Second, while the latent alignment model performs better than unstructured models, the difference diminishes after adding the enhanced lexical semantics information. This may suggest that compared to introducing complex structured constraints, incorporating shallow semantic information is both more effective and computationally inexpensive in improving the performance, at least for the specific word alignment model tested in this work.

The rest of the paper is structured as follows. We first survey the related work in Sec. 2. Sec. 3 defines the problem of answer sentence selection, along with the high-level description of our solution. The enhanced lexical semantic models and the learning frameworks we explore are presented in Sec. 4 and Sec. 5, respectively. Our experimental results on a benchmark QA dataset is shown in Sec. 6. Finally, Sec. 7 concludes the paper.

## 2 Related Work

While the task of question answering has a long history dated back to the dawn of artificial intelligence, early systems like STUDENT (Winograd, 1977) and LUNAR (Woods, 1973) are typically designed to demonstrate natural language understanding for a small and specific domain. The Text REtrieval Conference (TREC) Question Answering Track was arguably the first large-scale evaluation of open-domain question answering (Voorhees and Tice, 2000). The task is designed in an information retrieval oriented setting. Given a factoid question along with a collection of documents, a system is required to return the exact answer, along with the document that supports the answer. In contrast, the Jeopardy! TV

quiz show provides another open-domain question answering setting, in which IBM’s Watson system famously beat the two highest ranked players (Ferrucci, 2012). Questions in this game are presented in a statement form and the system needs to identify the true question and to give the exact answer. A short sentence or paragraph to justify the answer is not required in either TREC-QA or Jeopardy!

As any QA system can virtually be decomposed into two major high-level components, *retrieval* and *selection* (Echihabi and Marcu, 2003), the answer selection problem is clearly critical. Limiting the scope of an answer to a sentence is first highlighted by Wang et al. (2007), who argued that it was more informative to present the whole sentence instead of a short answer to users.

Observing the limitations of the bag-of-words models, Wang et al. (2007) proposed a syntax-driven approach, where each pair of question and sentence are matched by their dependency trees. The mapping is learned by a generative probabilistic model based on a Quasi-synchronous Grammar formulation (Smith and Eisner, 2006). This approach was later improved by Wang and Manning (2010) with a tree-edit CRF model that learns the latent alignment structure. In contrast, general tree matching methods based on tree-edit distance have been first proposed by Punyakanok et al. (2004) for a similar answer selection task. Heilman and Smith (2010) proposed a discriminative approach that first computes a tree kernel function between the dependency trees of the question and candidate sentence, and then learns a classifier based on the tree-edit features extracted.

Although lexical semantic information derived from WordNet has been used in some of these approaches, the research has mainly focused on modeling the mapping between the syntactic structures of questions and sentences, produced from syntactic analysis. The potential improvement from enhanced lexical semantic models seems to have been deliberately overlooked.<sup>1</sup>

## 3 Problem Definition

We consider the answer selection problem in a supervised learning setting. For a set of questions  $\{q_1, \dots, q_m\}$ , each question  $q_i$  is associated with a list of labeled candidate answer sentences

---

<sup>1</sup>For example, Heilman and Smith (2010) emphasized that “The tree edit model, which does not use lexical semantics knowledge, produced the best result reported to date.”

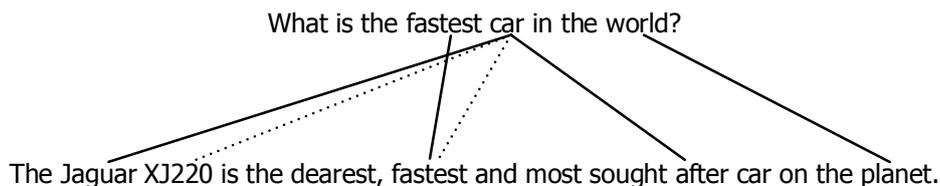


Figure 1: An example pair of question and answer sentence, adapted from (Harabagiu and Moldovan, 2001). Words connected by solid lines are clear synonyms or hyponym/hypernym; words with weaker semantic association are linked by dashed lines.

$\{(y_{i_1}, s_{i_1}), (y_{i_1}, s_{i_2}), \dots, (y_{i_n}, s_{i_n})\}$ , where  $y_{i_j} = 1$  indicates that sentence  $s_{i_j}$  is a correct answer to question  $q_i$ , and 0 otherwise. Using this labeled data, our goal is to learn a probabilistic classifier to predict the label of a new, unseen pair of question and sentence.

Fundamentally, what the classifier predicts is whether the sentence “matches” the question semantically. In other words, does  $s$  have the answer that satisfies the semantic constraints provided in the question? Without representing the question and sentence in logic or syntactic trees, we take a *word-alignment* view for solving this problem. We assume that there is an underlying structure  $h$  that describes how  $q$  and  $s$  can be associated through the relations of the words in them. Figure 1 illustrates this setting using a revised example from (Harabagiu and Moldovan, 2001). In this figure, words connected by solid lines are clear synonyms or hyponym/hypernym; words connected by dashed lines indicate that they are weakly related. With this alignment structure, features like the degree of mapping or whether all the content words in the question can be mapped to some words in the sentence can be extracted and help improve the classifier. Notice that the structure representation in terms of word-alignment is fairly general. For instance, if we assume a naive complete bipartite matching, then effectively it reduces to the simple bag-of-words model.

Typically, the “ideal” alignment structure is not available in the data, and previous work exploited mostly syntactic analysis (e.g., dependency trees) to reveal the latent mapping structure. In this work, we focus our study on leveraging the low-level semantic cues from recently proposed lexical semantic models. As will be shown in our experiments, such information not only improves a latent structure learning method, but also makes a simple

bipartite matching approach extremely strong.<sup>2</sup>

## 4 Lexical Semantic Models

In this section, we introduce the lexical semantic models we adopt for solving the semantic matching problem in answer selection. To go beyond the simple, limited surface-form matching, we aim to pair words that are semantically related, specifically measured by models of word relations including *synonymy/antonymy*, *hyponymy/hyponymy* (the Is-A relation) and general *semantic word similarity*.

### 4.1 Synonymy and Antonymy

Among all the word relations, *synonymy* is perhaps the most basic one and needs to be handled reliably. Although sets of synonyms can be easily found in thesauri or WordNet synsets, such resources typically cover only strict synonyms. When comparing two words, it is more useful to estimate the *degree* of synonymy as well. For instance, *ship* and *boat* are not strict synonyms because a ship is usually viewed as a large boat. Knowing that two words are somewhat synonymous could be valuable in determining whether they should be mapped.

In order to estimate the degree of synonymy, we leverage a recently proposed polarity-inducing latent semantic analysis (PILSA) model (Yih et al., 2012). Given a thesaurus, the model first constructs a *signed d-by-n* co-occurrence matrix  $W$ , where  $d$  is the number of word groups and  $n$  is the size of the vocabulary. Each row consists of a

<sup>2</sup>Proposed by an anonymous reviewer, one justification of this word-alignment approach, where syntactic analysis plays a less important role, is that there are often few sensible combinations of words. For instance, knowing only the set of words {“car”, “fastest”, “world”}, one may still guess correctly the question “What is the fastest car in the world?”

group of synonyms and antonyms of a particular sense and each column represents a unique word. Values of the elements in each row vector are the TFIDF values of the corresponding words in this group. The notion of *polarity* is then induced by making the values of words in the antonym groups negative, and the matrix is generalized by a low-rank approximation derived by singular-value decomposition (SVD) in the end. This design has an intriguing property – if the cosine score of two column vectors are positive, then the two corresponding words tend to be synonymous; if it's negative, then the two words are antonymous. The degree is measured by the absolute value.

Following the setting described in (Yih et al., 2012), we construct a PILSA model based on the Encarta thesaurus and enhance it with a discriminative projection matrix training method. The estimated degrees of both synonymy and antonymy are used our experiments.<sup>3</sup>

## 4.2 Hypernymy and Hyponymy

The *Class-Inclusion* or *Is-A* relation is commonly observed between words in questions and answer sentences. For example, to correctly answer the question “What color is Saturn?”, it is crucial that the selected sentence mentions a specific kind of color, as in “Saturn is a giant gas planet with brown and beige clouds.” Another example is “Who wrote Moonlight Sonata?”, where *compose* in “Ludwig van Beethoven composed the Moonlight Sonata in 1801.” is one kind of *write*.

Traditionally, WordNet taxonomy is the linguistic resource for identifying hypernyms and hyponyms, applied broadly to many NLP problems. However, WordNet has a number of well-known limitations including its rather limited or skewed concept distribution and the lack of the coverage of the *Is-A* relation (Song et al., 2011). For instance, when a word refers to a named entity, the particular sense and meaning is often not encoded. As a result, relations such as “Apple” *is-a* “company” and “Jaguar” *is-a* “car” cannot be found in WordNet. Similar to the case in synonymy, the *Is-A* relation defined in WordNet does not provide a native, real-valued degree of the relation, which can only be roughly approximated using the number of links on the taxonomy path connecting two

<sup>3</sup>Mapping two antonyms may be desired if one of them is in the scope of negation (Morante and Blanco, 2012; Blanco and Moldovan, 2011). However, we do not attempt to resolve the negation scope in this work.

concepts (Resnik, 1995).

In order to remedy these issues, we augment WordNet with the *Is-A* relations found in Probase (Wu et al., 2012). Probbase is a knowledge base that establishes connections between 2.7 million concepts, discovered automatically by applying Hearst patterns (Hearst, 1992) to 1.68 billion Web pages. Its abundant concept coverage distinguishes it from other knowledge bases, such as Freebase (Bollacker et al., 2008) and WikiTaxonomy (Ponzetto and Strube, 2007). Based on the frequency of term co-occurrences, each *Is-A* relation from Probbase is associated with a probability value, indicating the degree of the relation.

We verified the quality of Probbase *Is-A* relations using a recently proposed SemEval task of relational similarity (Jurgens et al., 2012) in a companion paper (Zhila et al., 2013), where a subset of the data is to measure the degree of two words having a *class-inclusion* relation. Probbase’s prediction correlates well with the human annotations and achieves a high Spearman’s rank correlation coefficient score,  $\rho = 0.619$ . In comparison, the previous best system (Rink and Harabagiu, 2012) in the task only reaches  $\rho = 0.233$ . These appealing qualities make Probbase a robust lexical semantic model for hypernymy/hyponymy.

## 4.3 Semantic Word Similarity

The third lexical semantic model we introduce targets a general notion of *word similarity*. Unlike synonymy and hyponymy, word similarity is only loosely defined when two words can be associated by some implicit relation.<sup>4</sup> The general word similarity model can be viewed as a “back-off” solution when the exact lexical relation (e.g., *part-whole* and *attribute*) is not available or cannot be accurately detected.

Among various word similarity models (Agirre et al., 2009; Reisinger and Mooney, 2010; Gabilovich and Markovitch, 2007; Radinsky et al., 2011), the vector space models (VSMs) based on the idea of *distributional similarity* (Turney and Pantel, 2010) are often used as the core component. Inspired by (Yih and Qazvinian, 2012), which argues the importance of incorporating heterogeneous vector space models for measuring word similarity, we leverage three different VSMs in this work: Wiki term-vectors, recurrent neural

<sup>4</sup>Instead of making the distinction, *word similarity* here refers to the larger set of relations commonly covered by *word relatedness* (Budanitsky and Hirst, 2006).

network language model (RNNLM) and a concept vector space model learned from click-through data. Semantic word similarity is estimated using the cosine score of the corresponding word vectors in these VSMs.

Contextual term-vectors created using the Wikipedia corpus have shown to perform well on measuring word similarity (Reisinger and Mooney, 2010). Following the setting suggested by Yih and Qazvinian (2012), we create term-vectors representing about 1 million words by aggregating terms within a window of  $[-10, 10]$  of each occurrence of the target word. The vectors are further refined by applying the same vocabulary and feature pruning techniques.

A recurrent neural network language model (Mikolov et al., 2010) aims to estimate the probability of observing a word given its preceding context. However, one by-product of this model is the word embedding learned in its hidden-layer, which can be viewed as capturing the word meaning in some latent, conceptual space. As a result, vectors of related words tend to be close to each other. For this word similarity model, we take a 640-dimensional version of RNNLM vectors, which is trained using the Broadcast News corpus of 320M words.<sup>5</sup>

The final word relatedness model is a projection model learned from the click-through data of a commercial search engine (Gao et al., 2011). Unlike the previous two models, which are created or trained using a text corpus, the input for this model is pairs of aggregated queries and titles of pages users click. This parallel data is used to train a projection matrix for creating the mapping between words in queries and documents based on user feedback, using a Siamese neural network (Yih et al., 2011). Each row vector of this matrix is the dense vector representation of the corresponding word in the vocabulary. Perhaps due to its unique information source, we found this particular word embedding seems to complement the other two VSMs and tends to improve the word similarity measure in general.

## 5 Learning QA Matching Models

In this section, we investigate the effectiveness of various learning models for matching questions and sentences, including the *bag-of-words* setting

<sup>5</sup><http://www.fit.vutbr.cz/~imikolov/rnnlm/>

and the framework of learning latent structures.

### 5.1 Bag-of-Words Model

The bag-of-words model treats each question and sentence as an unstructured bag of words. When comparing a question with a sentence, the model first matches each word in the question to each word in the sentence. It then aggregates features extracted from each of these word pairs to represent the whole question/sentence pair. A binary classifier can be trained easily using any machine learning algorithm in this standard supervised learning setting.

Formally, let  $x = (q, s)$  be a pair of question  $q$  and sentence  $s$ . Let  $V_q = \{w_{q_1}, w_{q_2}, \dots, w_{q_m}\}$  and  $V_s = \{w_{s_1}, w_{s_2}, \dots, w_{s_n}\}$  be the sets of words in  $q$  and  $s$ , respectively. Given a word pair  $(w_q, w_s)$ , where  $w_q \in V_q$  and  $w_s \in V_s$ , feature functions  $\phi_1, \dots, \phi_d$  map it to a  $d$ -dimensional real-valued feature vector.

We consider two aggregate functions for defining the feature vectors of the whole question/answer pair: *average* and *max*.

$$\Phi_{avg_j}(q, s) = \frac{1}{mn} \sum_{\substack{w_q \in V_q \\ w_s \in V_s}} \phi_j(w_q, w_s) \quad (1)$$

$$\Phi_{max_j}(q, s) = \max_{\substack{w_q \in V_q \\ w_s \in V_s}} \phi_j(w_q, w_s) \quad (2)$$

Together, each question/sentence pair is represented by a  $2d$ -dimensional feature vector.

We tested two learning algorithms in this setting: *logistic regression* and *boosted decision trees* (Friedman, 2001). The former is the log-linear model widely used in the NLP community and the latter is a robust non-linear learning algorithm that has shown great empirical performance.

The bag-of-words model does not require an additional inference stage as in structured learning, which may be computationally expensive. Nevertheless, its lack of structure information could limit the expressiveness of the model and make it difficult to capture more sophisticated semantics in the sentences. To address this concern, we investigate models of learning latent structures next.

### 5.2 Learning Latent Structures

One obvious issue of the bag-of-words model is that words in the unrelated part of the sentence may still be paired with words in the question, which introduces noise to the final feature vector.

This is observed in many question/sentence pairs, such as the one below.

**Q:** Which was the first movie that James Dean was in?

**A:** James Dean, who began as an actor on TV dramas, didn't make his screen debut until 1951's "Fixed Bayonet."

While this sentence correctly answers the question, the fact that James Dean began as a TV actor is unrelated to the question. As a result, an "ideal" word alignment structure should not link words in this clause to those in the question. In order to leverage the latent structured information, we adapt a recently proposed framework of *learning constrained latent representations* (LCLR) (Chang et al., 2010). LCLR can be viewed as a variant of Latent-SVM (Felzenszwalb et al., 2009) with different learning formulations and a general inference framework. The idea of LCLR is to replace the decision function of a standard linear model  $\theta^T \phi(x)$  with

$$\arg \max_h \theta^T \phi(x, h), \quad (3)$$

where  $\theta$  represents the weight vector and  $h$  represents the latent variables.

In this answer selection task,  $x = (q, s)$  represents a pair of question  $q$  and candidate sentence  $s$ . As described in Sec. 3,  $h$  refers to the latent alignment between  $q$  and  $s$ . The intuition behinds Eq. (3) is: candidate sentence  $s$  correctly answers question  $q$  if and only if the decision can be supported by the *best* alignment  $h$ .

The objective function of LCLR is defined as:

$$\begin{aligned} \min_{\theta} \quad & \frac{1}{2} \|\theta\|^2 + C \sum_i \xi_i^2 \\ \text{s.t.} \quad & \xi_i \geq 1 - y_i \max_h \theta^T \phi(x, h) \end{aligned}$$

Note that the alignment is latent, so LCLR uses the binary labels in the training data as feedback to find the alignment for each example.

The computational difficulty of the inference problem (Eq. (3)) largely depends on the constraints we enforce in the alignment. Complicated constraints may result in a difficult inference problem, which can be solved by integer linear programming (Roth and Yih, 2007). In this work, we considered several sets of constraints for the alignment task, including a two-layer phrase/word

alignment structure, but found that they generally performed the same. Therefore, we chose the many-to-one alignment<sup>6</sup>, where inference can be solved exactly using a simple greedy algorithm.

## 6 Experiments

We present our experimental results in this section by first introducing the data and evaluation metrics, followed by the results of existing systems and some baseline methods. We then show the positive impact of adding information of word relations from various lexical semantics models, with some discussion on the limitation of the word-matching approach.

### 6.1 Data & Evaluation Metrics

The answer selection dataset we used was originally created by Wang et al. (2007) based on the QA track of past Text REtrieval Conferences (TREC-QA). Questions in this dataset are short factoid questions, such as "What is Crips' gang color?" In average, each question is associated with approximately 33 answer candidate sentences. A pair of question and sentence is judged positive if the sentence contains the exact answer key *and* can provide sufficient context as supporting evidence.

The training set of the data contains manually labeled 5,919 question/sentence pairs from TREC 8-12. The development and testing sets are both from TREC 13, which contain 1,374 and 1,866 pairs, respectively. The task is treated as a sentence ranking problem for each question and thus evaluated in Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR), using the official TREC evaluation program. Following (Wang et al., 2007), candidate sentences with more than 40 words are removed from evaluation, as well as questions with only positive or negative candidate sentences.

### 6.2 Baseline Methods

Several systems have been proposed and tested using this dataset. Wang et al. (2007) presented a generative probabilistic model based on a Quasi-synchronous Grammar formulation and was later improved by Wang and Manning (2010) with a tree-edit CRF model that learns the latent alignment structure. In contrast, Heilman and

<sup>6</sup>Each word in the question needs to be linked to a word in the sentence. Each word in the sentence can be linked to zero or multiple words in the question.

System	MAP	MRR
Wang et al. (2007)	0.6029	0.6852
Heilman and Smith (2010)	0.6091	0.6917
Wang and Manning (2010)	0.5951	0.6951

Table 1: Test set results of existing methods, taken from Table 3 of (Wang and Manning, 2010).

Baseline	Dev		Test	
	MAP	MRR	MAP	MRR
Random	0.5243	0.5816	0.4708	0.5286
Word Cnt	0.6516	0.7216	0.6263	0.6822
Wgt Word Cnt	0.7112	0.7880	0.6531	0.7071

Table 2: Results of three baseline methods.

Smith (2010) proposed a discriminative approach that first computes a tree kernel function between the dependency trees of the question and candidate sentence, and then learns a classifier based on the tree-edit features extracted. Table 1 summarizes their results on the test set. All these systems incorporated lexical semantics features derived from WordNet and named entity features.

In order to further estimate the difficulty of this task and dataset, we tested three simple baselines. The first is *random scoring*, which simply assigns a random score to each candidate sentence. The second one, *word count*, is to count how many words in the question that also occur in the answer sentence, after removing stopwords<sup>7</sup>, and lowering the case. Finally, the last baseline method, *weighted word count*, is basically the same as identical word matching, but the count is re-weighted using the IDF value of the question word. This is similar to the BM25 ranking function (Robertson et al., 1995). The results of these three methods are shown in Table 1.

Somewhat surprisingly, we find that *word count* is fairly strong and performs comparably to previous systems.<sup>8</sup> In addition, weighting the question words with their IDF values further improves the results.

### 6.3 Incorporating Rich Lexical Semantics

We test the effectiveness of adding rich lexical semantics information by creating examples of different feature sets. As described in Sec. 5,

<sup>7</sup>We used a list of 101 stopwords, including articles, pronouns and punctuation.

<sup>8</sup>The finding has been confirmed by the lead author of (Wang et al., 2007).

all the features are based on the properties of the pair of a word from the question and a word from the candidate sentence. Stopwords are first removed from both questions and sentences and all words are lower-cased. Features used in the experiments can be categorized into six types: identical word matching (I), lemma matching (L), WordNet (WN), enhanced Lexical Semantics (LS), Named Entity matching (NE) and Answer type checking (Ans). Inspired by the *weighted word count* baseline, all features except (Ans) are weighted by the IDF value of the question word. In other words, the IDF values help decide the importance of word pairs to the model.

Starting from the our baseline model, *weighted word count*, the identical word matching (I) feature checks whether the pair of words are the same. Instead of checking the surface form of the word, lemma matching (L) verifies whether the two words have the same lemma form. Arguably the most common source of word relations, WordNet (WN) provides the primitive features of whether two words could belong to the same synset in WordNet, could be antonyms and whether one is a hypernym of the other. Alternatively, the enhanced lexical semantics (LS) features apply the models described in Sec. 4 to the word pair and use their estimated degree of synonymy, antonymy, hyponymy and semantic relatedness as features. Named entity matching (NE) checks whether two words are individually part of some named entities with the same type. Finally, when the question word is the WH-word, we check if the paired word belongs to some phrase that has the correct answer type using simple rules, such as “*Who* should link to a word that is part of a named entity of type *Person*.” We created examples in each round of experiments by augmenting these features in the same order, and observed how adding different information helped improve the model performance.

Three models are included in our study. For the unstructured, bag-of-words setting, we tested logistic regression (LR) and boosted decision trees (BDT). As mentioned in Sec. 5, the features for the whole question/sentence pair are the *average* and *max* of features of all the word pairs. For the structured-output setting, we used the framework of learning constrained latent representation (LCLR) and required that each question word needed to be mapped to a word in the sentence.

Feature set	LR		BDT		LCLR	
	MAP	MRR	MAP	MRR	MAP	MRR
1: I	0.6531	0.7071	0.6323	0.6898	0.6629	0.7279
2: I+L	0.6744	0.7223	0.6496	0.6923	0.6815	0.7270
3: I+L+WN	0.7039	0.7705	0.6798	0.7450	0.7316	0.7921
4: I+L+WN+LS	0.7339	0.8107	0.7523	0.8455	0.7626	0.8231
5: All	0.7374	0.8171	0.7495	0.8450	0.7648	0.8255

Table 3: Test results of various models and feature groups. Logistic regression (LR) and boosted decision trees (BDT) are the two unstructured models. LCLR is the algorithm for learning latent structures. Feature groups are identical word matching (I), lemma matching (L), WordNet (WN) and enhanced Lexical Semantics (LS). *All* includes these four plus Named Entity matching (NE) and Answer type checking (Ans).

Hyper-parameters are selected using the ones that achieve the best MAP score on the development set. Results of these models and feature sets are presented in Table 3.

We make two observations from the results. First, while incorporating more information of the word pairs in general helps, it is clear that mapping words beyond surface-form matching with the help of WordNet (Line #3 vs. #2) is important. Moreover, when richer information from other lexical semantic models is available, the performance can be further improved (Line #4 vs. #3). Overall, by simply incorporating more information on word relations, we gain approximately 10 points in both MAP and MRR compared to surface-form matching (Line #4 vs. #2), consistently across all three models. However, adding more information like named entity matching and answer type verification does not seem to help much (Line #5 vs. #4). Second, while the structured-output model usually performs better than both unstructured models (LCLR vs. LR & BDT), the performance gain diminishes after more information of word pairs is available (e.g., Lines #4 and #5).

#### 6.4 Limitation of Word Matching Models

Although we have demonstrated the benefits of leveraging various lexical semantic models to help find the association between words, the problem of question answering is nevertheless far from solved using the word-based approach. Examining the output of the LCLR model with all features on the development set, we found that there were three main sources of errors, including uncovered or inaccurate entity relations, the lack of robust question analysis and the need of high-level semantic

representation and inference. While the first two can be improved by, say, using a better named entity tagger, incorporating other knowledge bases and building a question classifier, how to solve the third problem is tricky. Below is an example:

**Q:** In what film is Gordon Gekko the main character?

**A:** He received a best actor Oscar in 1987 for his role as Gordon Gekko in “Wall Street”.

This is a correct answer sentence because “winning a best actor Oscar” implies that the role Gordon Gekko is the main character. It is hard to believe that a pure word-matching model would be able to solve this type of “inferential question answering” problem.

## 7 Conclusions

In this paper, we present an experimental study on solving the answer selection problem using enhanced lexical semantic models. Following the word-alignment paradigm, we find that the rich lexical semantic information improves the models consistently in the unstructured bag-of-words setting and also in the framework of learning latent structures. Another interesting finding we have is that while the latent structured model, LCLR, performs better than the other two unstructured models, the difference diminishes after more information, including the enhanced lexical semantic knowledge and answer type verification, has been incorporated. This may suggest that adding shallow semantic information is more effective than introducing complex structured constraints, at least for the specific word alignment model we experimented with in this work.

In the future, we plan to explore several directions. First, although we focus on improving TREC-style open-domain question answering in this work, we would like to apply the proposed technology to other QA scenarios, such as community-based QA (CQA). For instance, the sentence matching technique can help map a given question to some questions in an existing CQA database (e.g., Yahoo! Answers). Moreover, the answer sentence selection scheme could also be useful in extracting the most related sentences from the answer text to form a summary answer. Second, because the task of answer sentence selection is very similar to paraphrase detection (Dolan et al., 2004) and recognizing textual entailment (Dagan et al., 2006), we would like to investigate whether systems for these tasks can be improved by incorporating enhanced lexical semantic knowledge as well. Finally, we would like to improve our system for the answer sentence selection task and for question answering in general. In addition to following the directions suggested by the error analysis presented in Sec. 6.4, we plan to use logic-like semantic representations of questions and sentences, and explore the role of lexical semantics for handling questions that require inference.

## Acknowledgments

We are grateful to Mengqiu Wang for providing the dataset and helping clarify some issues in the experiments. We also thank Chris Burges and Hoi-fung Poon for valuable discussion and the anonymous reviewers for their useful comments.

## References

- E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca and A. Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of NAACL*, pages 19–27.
- M. Bilotti, P. Ogilvie, J. Callan, and E. Nyberg. 2007. Structured retrieval for question answering. In *Proceedings of SIGIR*, pages 351–358.
- E. Blanco and D. Moldovan. 2011. Semantic representation of negation using focus detection. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*.
- K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *ACM Conference on Management of Data (SIGMOD)*, pages 1247–1250.
- A. Budanitsky and G. Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32:13–47, March.
- M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. 2010. Discriminative learning over constrained latent representations. In *Proceedings of NAACL*.
- I. Dagan, O. Glickman, and B. Magnini, editors. 2006. *The PASCAL Recognising Textual Entailment Challenge*, volume 3944. Springer-Verlag, Berlin.
- W. Dolan, C. Quirk, and C. Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of COLING*.
- A. Echihabi and D. Marcu. 2003. A noisy-channel approach to question answering. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 16–23.
- Oren Etzioni. 2011. Search needs a shake-up. *Nature*, 476(7358):25–26.
- P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. 2009. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1).
- D. Ferrucci. 2012. Introduction to “This is Watson”. *IBM Journal of Research and Development*, 56(3.4):1–1.
- J. Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.
- E. Gabrilovich and S. Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *AAAI Conference on Artificial Intelligence (AAAI)*.
- J. Gao, K. Toutanova, and W. Yih. 2011. Clickthrough-based latent semantic models for web search. In *Proceedings of SIGIR*, pages 675–684.
- S. Harabagiu and D. Moldovan. 2001. Open-domain textual question answering. Tutorial of NAACL-2001.
- M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING*, pages 539–545.
- M. Heilman and N. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019.

- D. Jurgens, S. Mohammad, P. Turney, and K. Holyoak. 2012. SemEval-2012 Task 2: Measuring degrees of relational similarity. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 356–364.
- T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. 2010. Recurrent neural network based language model. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1045–1048.
- D. Moldovan, M. Paşca, S. Harabagiu, and M. Surdeanu. 2003. Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, 21(2):133–154.
- D. Moldovan, C. Clark, S. Harabagiu, and D. Hodges. 2007. COGEX: A semantically and contextually enriched logic prover for question answering. *Journal of Applied Logic*, 5(1):49–69.
- R. Morante and E. Blanco. 2012. \*SEM 2012 shared task: Resolving the scope and focus of negation. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 265–274.
- S. Ponzetto and M. Strube. 2007. Deriving a large scale taxonomy from wikipedia. In *AAAI Conference on Artificial Intelligence (AAAI)*.
- V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *International Symposium on Artificial Intelligence and Mathematics (AI & Math)*.
- K. Radinsky, E. Agichtein, E. Gabrilovich, and S. Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. In *WWW '11*, pages 337–346.
- J. Reisinger and R. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Proceedings of NAACL*.
- P. Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *International Joint Conference on Artificial Intelligence (IJCAI)*.
- B. Rink and S. Harabagiu. 2012. UTD: Determining relational similarity using lexical patterns. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 413–418.
- S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at TREC-3. In *Text REtrieval Conference (TREC)*, pages 109–109.
- D. Roth and W. Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.
- D. Shen and M. Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of EMNLP-CoNLL*, pages 12–21.
- D. Smith and J. Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the HLT-NAACL Workshop on Statistical Machine Translation*, pages 23–30.
- Y. Song, H. Wang, Z. Wang, H. Li, and W. Chen. 2011. Short text conceptualization using a probabilistic knowledgebase. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2330–2336.
- K. Tai. 1979. The tree-to-tree correction problem. *J. ACM*, 26(3):422–433, July.
- P. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.
- E. Voorhees and D. Tice. 2000. Building a question answering test collection. In *Proceedings of SIGIR*, pages 200–207.
- M. Wang and C. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of COLING*.
- M. Wang, N. Smith, and T. Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *Proceedings of EMNLP-CoNLL*.
- T. Winograd. 1977. Five lectures on artificial intelligence. In A. Zampolli, editor, *Linguistic Structures Processing*, pages 399–520. North Holland.
- W. Woods. 1973. Progress in natural language understanding: An application to lunar geology. In *Proceedings of the National Computer Conference and Exposition (AFIPS)*, pages 441–450.
- W. Wu, H. Li, H. Wang, and K. Zhu. 2012. Probase: a probabilistic taxonomy for text understanding. In *ACM Conference on Management of Data (SIGMOD)*, pages 481–492.
- W. Yih and V. Qazvinian. 2012. Measuring word relatedness using heterogeneous vector space models. In *Proceedings of NAACL-HLT 2012*, pages 616–620.
- W. Yih, K. Toutanova, J. Platt, and C. Meek. 2011. Learning discriminative projections for text similarity measures. In *ACL Conference on Natural Language Learning (CoNLL)*, pages 247–256.
- W. Yih, G. Zweig, and J. Platt. 2012. Polarity inducing latent semantic analysis. In *Proceedings of EMNLP-CoNLL*, pages 1212–1222.
- A. Zhila, W. Yih, C. Meek, G. Zweig, and T. Mikolov. 2013. Combining heterogeneous models for measuring relational similarity. In *Proceedings of HLT-NAACL*.

# Syntactic Patterns versus Word Alignment: Extracting Opinion Targets from Online Reviews

Kang Liu, Liheng Xu and Jun Zhao

National Laboratory of Pattern Recognition  
Institute of Automation, Chinese Academy of Sciences  
{kliu, lhxu, jzhao}@nlpr.ia.ac.cn

## Abstract

Mining opinion targets is a fundamental and important task for opinion mining from online reviews. To this end, there are usually two kinds of methods: syntax based and alignment based methods. Syntax based methods usually exploited syntactic patterns to extract opinion targets, which were however prone to suffer from parsing errors when dealing with online informal texts. In contrast, alignment based methods used word alignment model to fulfill this task, which could avoid parsing errors without using parsing. However, there is no research focusing on which kind of method is more better when given a certain amount of reviews. To fill this gap, this paper empirically studies how the performance of these two kinds of methods vary when changing the size, domain and language of the corpus. We further combine syntactic patterns with alignment model by using a partially supervised framework and investigate whether this combination is useful or not. In our experiments, we verify that our combination is effective on the corpus with small and medium size.

## 1 Introduction

With the rapid development of Web 2.0, huge amount of user reviews are springing up on the Web. Mining opinions from these reviews become more and more urgent since that customers expect to obtain fine-grained information of products and manufacturers need to obtain immediate feedbacks from customers. In opinion mining, extracting opinion targets is a basic subtask. It is to extract a list of the objects which users express their opinions on and can provide the prior information of targets for opinion mining. So this task

has attracted many attentions. To extract opinion targets, pervious approaches usually relied on opinion words which are the words used to express the opinions (Hu and Liu, 2004a; Popescu and Etzioni, 2005; Liu et al., 2005; Wang and Wang, 2008; Qiu et al., 2011; Liu et al., 2012). Intuitively, opinion words often appear around and modify opinion targets, and there are opinion relations and associations between them. If we have known some words to be opinion words, the words which those opinion words modify will have high probability to be opinion targets.

Therefore, identifying the aforementioned opinion relations between words is important for extracting opinion targets from reviews. To fulfill this aim, previous methods exploited the words co-occurrence information to indicate them (Hu and Liu, 2004a; Hu and Liu, 2004b). Obviously, these methods cannot obtain precise extraction because of the diverse expressions by reviewers, like long-span modified relations between words, etc. To handle this problem, several methods exploited syntactic information, where several heuristic patterns based on syntactic parsing were designed (Popescu and Etzioni, 2005; Qiu et al., 2009; Qiu et al., 2011). However, the sentences in online reviews usually have informal writing styles including grammar mistakes, typos, improper punctuation etc., which make parsing prone to generate mistakes. As a result, the syntax-based methods which heavily depended on the parsing performance would suffer from parsing errors (Zhang et al., 2010). To improve the extraction performance, we can only employ some exquisite high-precision patterns. But this strategy is likely to miss many opinion targets and has lower recall with the increase of corpus size. To resolve these problems, Liu et al. (2012) formulated identifying opinion relations between words as an monolingual alignment process. A word can find its corresponding modifiers by using a word alignment

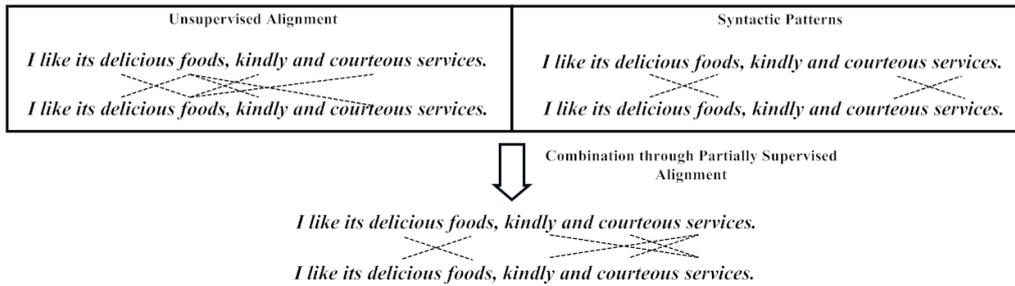


Figure 1: Mining Opinion Relations between Words using Partially Supervised Alignment Model

model (WAM). Without using syntactic parsing, the noises from parsing errors can be effectively avoided. Nevertheless, we notice that the alignment model is a statistical model which needs sufficient data to estimate parameters. When the data is insufficient, it would suffer from data sparseness and may make the performance decline.

Thus, from the above analysis, we can observe that the size of the corpus has impacts on these two kinds of methods, which arises some important questions: how can we make selection between syntax based methods and alignment based method for opinion target extraction when given a certain amount of reviews? And which kind of methods can obtain better extraction performance with the variation of the size of the dataset? Although (Liu et al., 2012) had proved the effectiveness of WAM, they mainly performed experiments on the dataset with medium size. We are still curious about that when the size of dataset is larger or smaller, can we obtain the same conclusion? To our best knowledge, these problems have not been studied before. Moreover, opinions may be expressed in different ways with the variation of the domain and language of the corpus. When the domain or language of the corpus is changed, what conclusions can we obtain? To answer these questions, in this paper, we adopt a unified framework to extract opinion targets from reviews, in the key component of which we vary the methods between syntactic patterns and alignment model. Then we run the whole framework on the corpus with different size (from #500 to #1,000,000), domain (three domains) and language (Chinese and English) to empirically assess the performance variations and discuss which method is more effective.

Furthermore, this paper naturally addresses another question: is it useful for opinion targets extraction when we combine syntactic patterns and word alignment model into a unified model? To

this end, we employ a partially supervised alignment model (PSWAM) like (Gao et al., 2010; Liu et al., 2013). Based on the exquisitely designed high-precision syntactic patterns, we can obtain some precisely modified relations between words in sentences, which provide a portion of links of the full alignments. Then, these partial alignment links can be regarded as the constrains for a standard unsupervised word alignment model. And each target candidate would find its modifier under the partial supervision. In this way, the errors generated in standard unsupervised WAM can be corrected. For example in Figure 1, “*kindly*” and “*courteous*” are incorrectly regarded as the modifiers for “*foods*” if the WAM is performed in an whole unsupervised framework. However, by using some high-precision syntactic patterns, we can assert “*courteous*” should be aligned to “*services*”, and “*delicious*” should be aligned to “*foods*”. Through combination under partial supervision, we can see “*kindly*” and “*courteous*” are correctly linked to “*services*”. Thus, it’s reasonable to expect to yield better performance than traditional methods. As mentioned in (Liu et al., 2013), using PSWAM can not only inherit the advantages of WAM: effectively avoiding noises from syntactic parsing errors when dealing with informal texts, but also can improve the mining performance by using partial supervision. However, is this kind of combination always useful for opinion target extraction? To access this problem, we also make comparison between PSWAM based method and the aforementioned methods in the same corpora with different size, language and domain. The experimental results show the combination by using PSWAM can be effective on dataset with small and medium size.

## 2 Related Work

Opinion target extraction isn't a new task for opinion mining. There are much work focusing on this task, such as (Hu and Liu, 2004b; Ding et al., 2008; Li et al., 2010; Popescu and Etzioni, 2005; Wu et al., 2009). Totally, previous studies can be divided into two main categories: supervised and unsupervised methods.

In supervised approaches, the opinion target extraction task was usually regarded as a sequence labeling problem (Jin and Huang, 2009; Li et al., 2010; Ma and Wan, 2010; Wu et al., 2009; Zhang et al., 2009). It's not only to extract a lexicon or list of opinion targets, but also to find out each opinion target mentions in reviews. Thus, the contextual words are usually selected as the features to indicate opinion targets in sentences. And classical sequence labeling models are used to train the extractor, such as CRFs (Li et al., 2010), HMM (Jin and Huang, 2009) etc.. Jin et al. (2009) proposed a lexicalized HMM model to perform opinion mining. Both Li et al. (2010) and Ma et al. (2010) used CRFs model to extract opinion targets in reviews. Specially, Li et al. proposed a Skip-Tree CRF model for opinion target extraction, which exploited three structures including linear-chain structure, syntactic structure, and conjunction structure. However, the main limitation of these supervised methods is the need of labeled training data. If the labeled training data is insufficient, the trained model would have unsatisfied extraction performance. Labeling sufficient training data is time and labor consuming. And for different domains, we need label data independently, which is obviously impracticable.

Thus, many researches focused on unsupervised methods, which are mainly to extract a list of opinion targets from reviews. Similar to ours, most approaches regarded opinion words as the indicator for opinion targets. (Hu and Liu, 2004a) regarded the nearest adjective to an noun/noun phrase as its modifier. Then it exploited an association rule mining algorithm to mine the associations between them. Finally, the frequent explicit product features can be extracted in a bootstrapping process by further combining item's frequency in dataset. Only using nearest neighbor rule to mine the modifier for each candidate cannot obtain precise results. Thus, (Popescu and Etzioni, 2005) used syntax information to extract opinion targets, which designed some syntactic patterns to capture

the modified relations between words. The experimental results showed that their method had better performance than (Hu and Liu, 2004a). Moreover, (Qiu et al., 2011) proposed a Double Propagation method to expand sentiment words and opinion targets iteratively, where they also exploited syntactic relations between words. Specially, (Qiu et al., 2011) didn't only design syntactic patterns for capturing modified relations, but also designed patterns for capturing relations among opinion targets and relations among opinion words. However, the main limitation of Qiu's method is that the patterns based on dependency parsing tree may miss many targets for the large corpora. Therefore, Zhang et al. (2010) extended Qiu's method. Besides the patterns used in Qiu's method, they adopted some other special designed patterns to increase recall. In addition they used the HITS (Kleinberg, 1999) algorithm to compute opinion target confidences to improve the precision. (Liu et al., 2012) formulated identifying opinion relations between words as an alignment process. They used a completely unsupervised WAM to capture opinion relations in sentences. Then the opinion targets were extracted in a standard random walk framework where two factors were considered: opinion relevance and target importance. Their experimental results have shown that WAM was more effective than traditional syntax-based methods for this task. (Liu et al., 2013) extend Liu's method, which is similar to our method and also used a partially supervised alignment model to extract opinion targets from reviews. We notice these two methods ((Liu et al., 2012) and (Liu et al., 2013)) only performed experiments on the corpora with a medium size. Although both of them proved that WAM model is better than the methods based on syntactic patterns, they didn't discuss the performance variation when dealing with the corpora with different sizes, especially when the size of the corpus is less than 1,000 and more than 10,000. Based on their conclusions, we still don't know which kind of methods should be selected for opinion target extraction when given a certain amount of reviews.

## 3 Opinion Target Extraction Methodology

To extract opinion targets from reviews, we adopt the framework proposed by (Liu et al., 2012), which is a graph-based extraction framework and

has two main components as follows.

1) **The first component** is to capture opinion relations in sentences and estimate associations between opinion target candidates and potential opinion words. In this paper, we assume opinion targets to be nouns or noun phrases, and opinion words may be adjectives or verbs, which are usually adopted by (Hu and Liu, 2004a; Qiu et al., 2011; Wang and Wang, 2008; Liu et al., 2012). And a potential opinion relation is comprised of an opinion target candidate and its corresponding modified word.

2) **The second component** is to estimate the confidence of each candidate. The candidates with higher confidence scores than a threshold will be extracted as opinion targets. In this procedure, we formulate the associations between opinion target candidates and potential opinion words in a bipartite graph. A random walk based algorithm is employed on this graph to estimate the confidence of each target candidate.

In this paper, we fix the method in the second component and vary the algorithms in the first component. In the first component, we respectively use syntactic patterns and unsupervised word alignment model (WAM) to capture opinion relations. In addition, we employ a partially supervised word alignment model (PSWAM) to incorporate syntactic information into WAM. In experiments, we run the whole framework on the different corpora to discuss which method is more effective. In the following subsections, we will present them in detail.

### 3.1 The First Component: Capturing Opinion Relations and Estimating Associations between Words

#### 3.1.1 Syntactic Patterns

To capture opinion relations in sentences by using syntactic patterns, we employ the manual designed syntactic patterns proposed by (Qiu et al., 2011). Similar to Qiu, only the syntactic patterns based on the direct dependency are employed to guarantee the extraction qualities. The direct dependency has two types. The first type indicates that one word depends on the other word without any additional words in their dependency path. The second type denotes that two words both depend on a third word directly. Specifically, we employ Minipar<sup>1</sup> to parse sentences. To further make syn-

tactic patterns precisely, we only use a few dependency relation labels outputted by Minipar, such as mod, pnm, subj, desc etc. To make a clear explanation, we give out some syntactic pattern examples in Table 1. In these patterns, *OC* is a potential opinion word which is an adjective or a verb. *TC* is an opinion target candidate which is a noun or noun phrase. The item on the arrows means the dependency relation type. The item in parenthesis denotes the part-of-speech of the other word. In these examples, the first three patterns are based on the first direct dependency type and the last two patterns are based on the second direct dependency type.

Pattern#1: <OC> $\xrightarrow{mod}$ <TC>
Example: This phone has an <i>amazing design</i>
Pattern#2: <TC> $\xrightarrow{obj}$ <OC>
Example: I <i>like</i> this <i>phone</i> very much
Pattern#3: <OC> $\xrightarrow{pnm}$ <TC>
Example: the <i>buttons</i> <i>easier</i> to use
Pattern#4: <OC> $\xrightarrow{mod}$ (NN) $\xleftarrow{subj}$ <TC>
Example: <i>IPhone</i> is a <i>revolutionary</i> smart phone
Pattern#5: <OC> $\xrightarrow{pred}$ (VBE) $\xleftarrow{subj}$ <TC>
Example: The <i>quality</i> of LCD is <i>good</i>

Table 1: Some Examples of Used Syntactic Patterns

#### 3.1.2 Unsupervised Word Alignment Model

In this subsection, we present our method for capturing opinion relations using unsupervised word alignment model. Similar to (Liu et al., 2012), every sentence in reviews is replicated to generate a parallel sentence pair, and the word alignment algorithm is applied to the monolingual scenario to align a noun/noun phrase with its modifiers. We select IBM-3 model (Brown et al., 1993) as the alignment model. Formally, given a sentence  $S = \{w_1, w_2, \dots, w_n\}$ , we have

$$P_{ibm3}(A|S) \propto \prod_{i=1}^N n(\phi_i|w_i) \prod_{j=1}^N t(w_j|w_{a_j}) d(j|a_j, N) \quad (1)$$

where  $t(w_j|w_{a_j})$  models the co-occurrence information of two words in dataset.  $d(j|a_j, N)$  models word position information, which describes the probability of a word in position  $a_j$  aligned with a word in position  $j$ . And  $n(\phi_i|w_i)$  describes the ability of a word for modifying (being modified by) several words.  $\phi_i$  denotes the number of words

<sup>1</sup><http://webdocs.cs.ualberta.ca/lindek/minipar.htm>

that are aligned with  $w_i$ . In our experiments, we set  $\phi_i = 2$ .

Since we only have interests on capturing opinion relations between words, we only pay attentions on the alignments between opinion target candidates (nouns/noun phrases) and potential opinion words (adjectives/verbs). If we directly use the alignment model, a noun (noun phrase) may align with other unrelated words, like prepositions or conjunctions and so on. Thus, we set constrains on the model: 1) Alignment links must be assigned among nouns/noun phrases, adjectives/verbs and null words. Aligning to null words means that this word has no modifier or modifies nothing; 2) Other unrelated words can only align with themselves.

### 3.1.3 Combining Syntax-based Method with Alignment-based Method

In this subsection, we try to combine syntactic information with word alignment model. As mentioned in the first section, we adopt a partially supervised alignment model to make this combination. Here, the opinion relations obtained through the high-precision syntactic patterns (Section 3.1.1) are regarded as the ground truth and can only provide a part of full alignments in sentences. They are treated as the constrains for the word alignment model. Given some partial alignment links  $\hat{A} = \{(k, a_k) | k \in [1, n], a_k \in [1, n]\}$ , the optimal word alignment  $A^* = \{(i, a_i) | i \in [1, n], a_i \in [1, n]\}$  can be obtained as  $A^* = \underset{A}{\operatorname{argmax}} P(A|S, \hat{A})$ , where  $(i, a_i)$  means that a noun (noun phrase) at position  $i$  is aligned with its modifier at position  $a_i$ .

Since the labeled data provided by syntactic patterns is not a full alignment, we adopt a EM-based algorithm, named as constrained hill-climbing algorithm (Gao et al., 2010), to estimate the parameters in the model. In the training process, the constrained hill-climbing algorithm can ensure that the final model is marginalized on the partial alignment links. Particularly, in the E step, their method aims to find out the alignments which are consistent to the alignment links provided by syntactic patterns, where there are main two steps involved.

1) *Optimize towards the constraints.* This step aims to generate an initial alignments for alignment model (IBM-3 model in our method), which can be close to the constraints. First, a simple alignment model (IBM-1, IBM-2, HMM etc.) is

trained. Then, the evidence being inconsistent to the partial alignment links will be got rid of by using the move operator  $m_{i,j}$  which changes  $a_j = i$  and the swap operator  $s_{j_1, j_2}$  which exchanges  $a_{j_1}$  and  $a_{j_2}$ . The alignment is updated iteratively until no additional inconsistent links can be removed.

2) *Towards the optimal alignment under the constraints.* This step aims to optimize towards the optimal alignment under the constraints which starts from the aforementioned initial alignments. Gao et.al. (2010) set the corresponding cost value of the invalid move or swap operation in  $M$  and  $S$  to be negative, where  $M$  and  $S$  are respectively called Moving Matrix and Swapping Matrix, which record all possible move and swap costs between two different alignments. In this way, the invalid operators will never be picked which can guarantee that the final alignment links to have high probability to be consistent with the partial alignment links provided by high-precision syntactic patterns.

Then in M-step, evidences from the neighbor of final alignments are collected so that we can produce the estimation of parameters for the next iteration. In the process, those statistics which come from inconsistent alignment links aren't be picked up. Thus, we have

$$P(w_i | w_{a_i}, \hat{A}) = \begin{cases} \lambda, & \text{otherwise} \\ P(w_i | w_{a_i}) + \lambda, & \text{inconsistent with } \hat{A} \end{cases} \quad (2)$$

where  $\lambda$  means that we make soft constraints on the alignment model. As a result, we expect some errors generated through high-precision patterns (Section 3.1.1) may be revised in the alignment process.

## 3.2 Estimating Associations between Words

After capturing opinion relations in sentences, we can obtain a lot of word pairs, each of which is comprised of an opinion target candidate and its corresponding modified word. Then the conditional probabilities between potential opinion target  $w_t$  and potential opinion word  $w_o$  can be estimated by using maximum likelihood estimation. Thus, we have  $P(w_t | w_o) = \frac{\operatorname{Count}(w_t, w_o)}{\operatorname{Count}(w_o)}$ , where  $\operatorname{Count}(\cdot)$  means the item's frequency information.  $P(w_t | w_o)$  means the conditional probabilities between two words. At the same time, we can obtain conditional probability  $P(w_o | w_t)$ . Then,

similar to (Liu et al., 2012), the association between an opinion target candidate and its modifier is estimated as follows.  $Association(w_t, w_o) = (\alpha \times P(w_t|w_o) + (1 - \alpha) \times P(w_o|w_t))^{-1}$ , where  $\alpha$  is the harmonic factor. We set  $\alpha = 0.5$  in our experiments.

### 3.3 The Second Component: Estimating Candidate Confidence

In the second component, we adopt a graph-based algorithm used in (Liu et al., 2012) to compute the confidence of each opinion target candidate, and the candidates with higher confidence than the threshold will be extracted as the opinion targets. Here, opinion words are regarded as the important indicators. We assume that two target candidates are likely to belong to the similar category, if they are modified by similar opinion words. Thus, we can propagate the opinion target confidences through opinion words.

To model the mined associations between words, a bipartite graph is constructed, which is defined as a weighted undirected graph  $G = (V, E, W)$ . It contains two kinds of vertex: opinion target candidates and potential opinion words, respectively denoted as  $v_t \in V$  and  $v_o \in V$ . As shown in Figure 2, the white vertices represent opinion target candidates and the gray vertices represent potential opinion words. An edge  $e_{v_t, v_o} \in E$  between vertices represents that there is an opinion relation, and the weight  $w$  on the edge represents the association between two words.

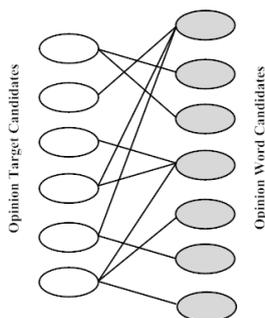


Figure 2: Modeling Opinion Relations between Words in a Bipartite Graph

To estimate the confidence of each opinion target candidate, we employ a random walk algorithm on our graph, which iteratively computes the weighted average of opinion target confidences from neighboring vertices. Thus we have

$$C^{i+1} = (1 - \beta) \times M \times M^T \times C^i + \beta \times I \quad (3)$$

where  $C^{i+1}$  and  $C^i$  respectively represent the opinion target confidence vector in the  $(i + 1)^{th}$  and  $i^{th}$  iteration.  $M$  is the matrix of word associations, where  $M_{i,j}$  denotes the association between the opinion target candidate  $i$  and the potential opinion word  $j$ . And  $I$  is defined as the prior confidence of each candidate for opinion target. Similar to (Liu et al., 2012), we set each item in  $I_v = \frac{tf(v)idf(v)}{\sum_v tf(v)idf(v)}$ , where  $tf(v)$  is the term frequency of  $v$  in the corpus, and  $idf(v)$  is computed by using the Google n-gram corpus<sup>2</sup>.  $\beta \in [0, 1]$  represents the impact of candidate prior knowledge on the final estimation results. In experiments, we set  $\beta = 0.4$ . The algorithm run until convergence which is achieved when the confidence on each node ceases to change in a tolerance value.

## 4 Experiments

### 4.1 Datasets and Evaluation Metrics

In this section, to answer the questions mentioned in the first section, we collect a large collection named as *LARGE*, which includes reviews from three different domains and different languages. This collection was also used in (Liu et al., 2012). In the experiments, reviews are first segmented into sentences according to punctuation. The detailed statistical information of the used collection is shown in Table 2, where Restaurant is crawled from the Chinese Web site: [www.dianping.com](http://www.dianping.com). The Hotel and MP3 are used in (Wang et al., 2011), which are respectively crawled from [www.tripadvisor.com](http://www.tripadvisor.com) and [www.amazon.com](http://www.amazon.com). For each dataset, we perform random sampling to generate testing set with different sizes, where we use sampled subsets with  $\#sentences = 5 \times 10^2, 10^3, 5 \times 10^3, 10^4, 5 \times 10^4, 10^5$  and  $10^6$  sentences respectively. Each

Domain	Language	Sentence	Reviews
<b>Restaurant</b>	Chinese	1,683,129	395,124
<b>Hotel</b>	English	1,855,351	185,829
<b>MP3</b>	English	289,931	30,837

Table 2: Experimental Dataset

sentence is tokenized, part-of-speech tagged by using Stanford NLP tool<sup>3</sup>, and parsed by using Minipar toolkit. And the method of (Zhu et al., 2009) is used to identify noun phrases.

<sup>2</sup><http://books.google.com/ngrams/datasets>

<sup>3</sup><http://nlp.stanford.edu/software/tagger.shtml>

We select precision and recall as the metrics. Specifically, to obtain the ground truth, we manually label all opinion targets for each subset. In this process, three annotators are involved. First, every noun/noun phrase and its contexts in review sentences are extracted. Then two annotators were required to judge whether every noun/noun phrase is opinion target or not. If a conflict happens, a third annotator will make judgment for final results. The average inter-agreements is 0.74. We also perform a significant test, i.e., a t-test with a default significant level of 0.05.

## 4.2 Compared Methods

We select three methods for comparison as follows.

- **Syntax**: It uses syntactic patterns mentioned in Section 3.1.1 in the first component to capture opinion relations in reviews. Then the associations between words are estimated and the graph based algorithm proposed in the second component (Section 3.3) is performed to extract opinion targets.
- **WAM**: It is similar to **Syntax**, where the only difference is that **WAM** uses unsupervised WAM (Section 3.1.2) to capture opinion relations.
- **PSWAM** is similar to **Syntax** and **WAM**, where the difference is that **PSWAM** uses the method mentioned in Section 3.1.3 to capture opinion relations, which incorporates syntactic information into word alignment model by using partially supervised framework.

The experimental results on different domains are respectively shown in Figure 3, 4 and 5.

## 4.3 Syntax based Methods vs. Alignment based Methods

Comparing **Syntax** with **WAM** and **PSWAM**, we can obtain the following observations:

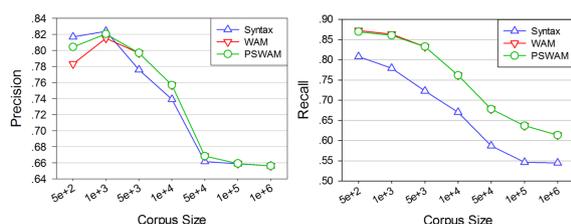


Figure 3: Experimental results on Restaurant

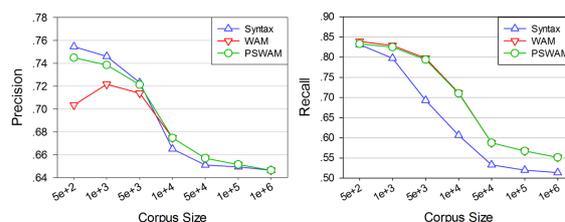


Figure 4: Experimental results on Hotel

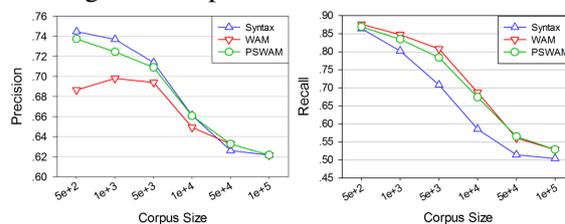


Figure 5: Experimental results on MP3

1) When the size of the corpus is small, **Syntax** has better precision than alignment based methods (**WAM** and **PSWAM**). We believe the reason is that the high-precision syntactic patterns employed in **Syntax** can effectively capture opinion relations in a small amount of texts. In contrast, the methods based on word alignment model may suffer from data sparseness for parameter estimation, so the precision is lower.

2) However, when the size of the corpus increases, the precision of **Syntax** decreases, even worse than alignment based methods. We believe it's because more noises were introduced from parsing errors with the increase of the size of the corpus, which will have more negative impacts on extraction results. In contrast, for estimating the parameters of alignment based methods, the data is more sufficient, so the precision is better compared with syntax based method.

3) We also observe that recall of **Syntax** is worse than other two methods. It's because the human expressions of opinions are diverse and the manual designed syntactic patterns are limited to capture all opinion relations in sentences, which may miss an amount of correct opinion targets.

4) It's interesting that the performance gap between these three methods is smaller with the increase of the size of the corpus (more than 50,000). We guess the reason is that when the data is sufficient enough, we can obtain sufficient statistics for each opinion target. In such situation, the graph-based ranking algorithm in the second component will be apt to be affected by the frequency information, so the final performance could not be sensitive to the performance of opinion relations iden-

tification in the first component. Thus, in this situation, we can get conclusion that there is no obviously difference on performance between syntax-based approach and alignment-based approach.

5) From the results on dataset with different languages and different domains, we can obtain the similar observations. It indicates that choosing either syntactic patterns or word alignment model for extracting opinion targets can take a few consideration on the language and domain of the corpus.

Thus, based on the above observations, we can draw the following conclusions: making chooses between different methods is only related to the size of the corpus. The method based on syntactic patterns is more suitable for small corpus ( $\#sentences < 5 \times 10^3$  shown in our experiments). And word alignment model is more suitable for medium corpus ( $5 \times 10^3 < \#sentences < 5 \times 10^4$ ). Moreover, when the size of the corpus is big enough, the performance of two kinds of methods tend to become the same ( $\#sentences \geq 10^5$  shown in our experiments).

#### 4.4 Is It Useful Combining Syntactic Patterns with Word Alignment Model

In this subsection, we try to see whether combining syntactic information with alignment model by using PSWAM is effective or not for opinion target extraction. From the results in Figure 3, 4 and 5, we can see that **PSWAM** has the similar recall compared with **WAM** in all datasets. **PSWAM** outperforms **WAM** on precision in all dataset. But the precision gap between **PSWAM** and **WAM** decreases when the size of the corpus increases. When the size is larger than  $5 \times 10^4$ , the performance of these two methods is almost the same. We guess the reason is that more noises from parsing errors will be introduced by syntactic patterns with the increase of the size of corpus, which have negative impacts on alignment performance. At the same time, as mentioned above, a great deal of reviews will bring sufficient statistics for estimating parameters in alignment model, so the roles of partial supervision from syntactic information will be covered by frequency information used in our graph based ranking algorithm.

**Compared with State-of-the-art Methods.** However, it's not say that this combination is not useful. From the results, we still see that **PSWAM** outperforms **WAM** in all datasets on

precision when size of corpus is smaller than  $5 \times 10^4$ . To further prove the effectiveness of our combination, we compare **PSWAM** with some state-of-the-art methods, including **Hu** (Hu and Liu, 2004a), which extracted frequent opinion target words based on association mining rules, **DP** (Qiu et al., 2011), which extracted opinion targets through syntactic patterns, and **LIU** (Liu et al., 2012), which fulfilled this task by using unsupervised WAM. The parameter settings in these baselines are the same as the settings in the original papers. Because of the space limitation, we only show the results on Restaurant and Hotel, as shown in Figure 6 and 7.

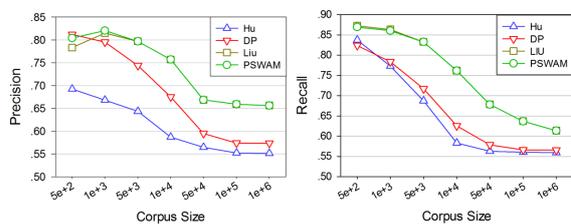


Figure 6: Compared with the State-of-the-art Methods on Restaurant

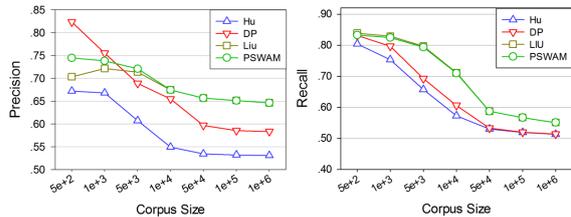


Figure 7: Compared with the State-of-the-art Methods on Hotel

From the experimental results, we can obtain the following observations. **PSWAM** outperforms other methods in most datasets. This indicates that our method based on PSWAM is effective for opinion target extraction. Especially compared **PSWAM** with **LIU**, both of which are based on word alignment model, we can see **PSWAM** identifies opinion relations by performing WAM under partial supervision, which can effectively improve the precision when dealing with small and medium corpus. However, these improvements are limited when the size of the corpus increases, which has the similar observations obtained above.

**The Impact of Syntactic Information on Word Alignment Model.** Although we have prove the effectiveness of PSWAM in the corpus with small and medium size, we are still curious about how the performance varies when we incor-

porate different amount of syntactic information into WAM. In this experiment, we rank the used syntactic patterns mentioned in Section 3.1.1 according to the quantities of the extracted alignment links by these patterns. Then, to capture opinion relations, we respectively use top N syntactic patterns according to frequency mentioned above to generate partial alignment links for PSWAM in section 3.1.3. We respectively define  $N=[1,7]$ . The larger is N, the more syntactic information is incorporated. Because of the space limitation, only the average performance of all dataset is shown in Figure 8.

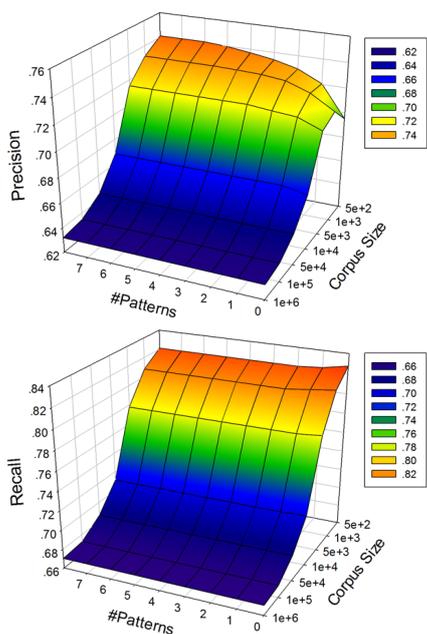


Figure 8: The Impacts of Different Syntactic Information on Word Alignment Model

In Figure 8, we can observe that the syntactic information mainly have effect on precision. When the size of the corpus is small, the opinion relations mined by high-precision syntactic patterns are usually correct, so incorporating more syntactic information can improve the precision of word alignment model more. However, when the size of the corpus increases, incorporating more syntactic information has little impact on precision.

## 5 Conclusions and Future Work

This paper discusses the performance variation of syntax based methods and alignment based methods on opinion target extraction task for the dataset with different sizes, different languages and different domains. Through experimental results, we can see that choosing which method is not related

with corpus domain and language, but strongly associated with the size of the corpus. We can conclude that syntax-based method is likely to be more effective when the size of the corpus is small, and alignment-based methods are more useful for the medium size corpus. We further verify that incorporating syntactic information into word alignment model by using PSWAM is effective when dealing with the corpora with small or medium size. When the size of the corpus is larger and larger, the performance gap between syntax based, WAM and PSWAM will decrease.

In future work, we will extract opinion targets based on not only opinion relations. Other semantic relations, such as the topical associations between opinion targets (or opinion words) should also be employed. We believe that considering multiple semantic associations will help to improve the performance. In this way, how to model heterogenous relations in a unified model for opinion targets extraction is worthy to be studied.

## Acknowledgement

This work was supported by the National Natural Science Foundation of China (No. 61070106, No. 61272332 and No. 61202329), the National High Technology Development 863 Program of China (No. 2012AA011102), the National Basic Research Program of China (No. 2012CB316300), Tsinghua National Laboratory for Information Science and Technology (TNList) Cross-discipline Foundation and the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (ICDD201201).

## References

- Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19(2):263–311, June.
- Xiaowen Ding, Bing Liu, and Philip S. Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the Conference on Web Search and Web Data Mining (WSDM)*.
- Qin Gao, Nguyen Bach, and Stephan Vogel. 2010. A semi-supervised word alignment algorithm with partial manual alignments. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 1–10, Uppsala, Sweden, July. Association for Computational Linguistics.

- Mingqin Hu and Bing Liu. 2004a. Mining opinion features in customer reviews. In *Proceedings of Conference on Artificial Intelligence (AAAI)*.
- Mingqing Hu and Bing Liu. 2004b. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '04*, pages 168–177, New York, NY, USA. ACM.
- Wei Jin and Hay Ho Huang. 2009. A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of International Conference on Machine Learning (ICML)*.
- Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September.
- Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Yingju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING*, pages 653–661. Tsinghua University Press.
- Bing Liu, Mingqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In Allan Ellis and Tatsuya Hagino, editors, *WWW*, pages 342–351. ACM.
- Kang Liu, Liheng Xu, and Jun Zhao. 2012. Opinion target extraction using word-based translation model. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1346–1356, Jeju Island, Korea, July. Association for Computational Linguistics.
- Kang Liu, Liheng Xu, Yang Liu, and Jun Zhao. 2013. Opinion target extraction using partially supervised word alignment model.
- Tengfei Ma and Xiaojun Wan. 2010. Opinion target extraction in chinese news comments. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING (Posters)*, pages 782–790. Chinese Information Processing Society of China.
- Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05*, pages 339–346, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Che. 2009. Expanding domain sentiment lexicon through double propagation.
- Guang Qiu, Bing Liu 0001, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37(1):9–27.
- Bo Wang and Houfeng Wang. 2008. Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing.
- Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In Chid Apt, Joydeep Ghosh, and Padhraic Smyth, editors, *KDD*, pages 618–626. ACM.
- Yuanbin Wu, Qi Zhang, Xuanjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *EMNLP*, pages 1533–1541. ACL.
- Qi Zhang, Yuanbin Wu, Tao Li, Mitsunori Ogihara, Joseph Johnson, and Xuanjing Huang. 2009. Mining product reviews based on shallow dependency parsing. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, SIGIR '09*, pages 726–727, New York, NY, USA. ACM.
- Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING (Posters)*, pages 1462–1470. Chinese Information Processing Society of China.
- Jingbo Zhu, Huizhen Wang, Benjamin K. Tsou, and Muhua Zhu. 2009. Multi-aspect opinion polling from textual reviews. In David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy J. Lin, editors, *CIKM*, pages 1799–1802. ACM.

# Mining Opinion Words and Opinion Targets in a Two-Stage Framework

Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen and Jun Zhao

National Laboratory of Pattern Recognition

Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China

{lhxu, kliu, swlai, ybchen, jzhao}@nlpr.ia.ac.cn

## Abstract

This paper proposes a novel two-stage method for mining opinion words and opinion targets. In the first stage, we propose a *Sentiment Graph Walking* algorithm, which naturally incorporates syntactic patterns in a Sentiment Graph to extract opinion word/target candidates. Then random walking is employed to estimate confidence of candidates, which improves extraction accuracy by considering confidence of patterns. In the second stage, we adopt a self-learning strategy to refine the results from the first stage, especially for filtering out high-frequency noise terms and capturing the long-tail terms, which are not investigated by previous methods. The experimental results on three real world datasets demonstrate the effectiveness of our approach compared with state-of-the-art unsupervised methods.

## 1 Introduction

Opinion mining not only assists users to make informed purchase decisions, but also helps business organizations understand and act upon customer feedbacks on their products or services in real-time. Extracting opinion words and opinion targets are two key tasks in opinion mining. Opinion words refer to those terms indicating positive or negative sentiment. Opinion targets represent aspects or attributes of objects toward which opinions are expressed. Mining these terms from reviews of a specific domain allows a more thorough understanding of customers' opinions.

Opinion words and opinion targets often co-occur in reviews and there exist modified relations (called *opinion relation* in this paper) between them. For example, in the sentence “It has a clear screen”, “clear” is an opinion word and “screen” is

an opinion target, and there is an opinion relation between the two words. It is natural to identify such opinion relations through common syntactic patterns (also called *opinion patterns* in this paper) between opinion words and targets. For example, we can extract “clear” and “screen” by using a syntactic pattern “Adj-*{mod}*-Noun”, which captures the opinion relation between them. Although previous works have shown the effectiveness of syntactic patterns for this task (Qiu et al., 2009; Zhang et al., 2010), they still have some limitations as follows.

**False Opinion Relations:** As an example, the phrase “everyday at school” can be matched by a pattern “Adj-*{mod}*-(Prep)-*{pcomp-n}*-Noun”, but it doesn't bear any sentiment orientation. We call such relations that match opinion patterns but express no opinion *false opinion relations*. Previous pattern learning algorithms (Zhuang et al., 2006; Kessler and Nicolov, 2009; Jijkoun et al., 2010) often extract opinion patterns by frequency. However, some high-frequency syntactic patterns can have very poor precision (Kessler and Nicolov, 2009).

**False Opinion Targets:** In another case, the phrase “wonderful time” can be matched by an opinion pattern “Adj-*{mod}*-Noun”, which is widely used in previous works (Popescu and Etzioni, 2005; Qiu et al., 2009). As can be seen, this phrase does express a positive opinion but unfortunately “time” is not a valid opinion target for most domains such as MP3. Thus, *false opinion targets* are extracted. Due to the lack of ground-truth knowledge for opinion targets, non-target terms introduced in this way can be hardly filtered out.

**Long-tail Opinion Targets:** We further notice that previous works prone to extract opinion targets with high frequency (Hu and Liu, 2004; Popescu and Etzioni, 2005; Qiu et al., 2009; Zhu et al., 2009), and they often have difficulty in identifying the infrequent or *long-tail opinion targets*.

To address the problems stated above, this paper proposes a two-stage framework for mining opinion words and opinion targets. The underlying motivation is analogous to the novel idea “Mine the Easy, Classify the Hard” (Dasgupta and Ng, 2009). In our first stage, we propose a *Sentiment Graph Walking* algorithm to cope with the false opinion relation problem, which mines easy cases of opinion words/targets. We speculate that it may be helpful to introduce a confidence score for each pattern. Concretely, we create a *Sentiment Graph* to model opinion relations among opinion word/target/pattern candidates and apply random walking to estimate confidence of them. Thus, confidence of pattern is considered in a unified process. Patterns that often extract false opinion relations will have low confidence, and terms introduced by low-confidence patterns will also have low confidence accordingly. This could potentially improve the extraction accuracy.

In the second stage, we identify the hard cases, which aims to filter out false opinion targets and extract long-tail opinion targets. Previous supervised methods have been shown to achieve state-of-the-art results for this task (Wu et al., 2009; Jin and Ho, 2009; Li et al., 2010). However, the big challenge for fully supervised method is the lack of annotated training data. Therefore, we adopt a **self-learning strategy**. Specifically, we employ a semi-supervised classifier to refine the target results from the first stage, which uses some highly confident target candidates as the initial labeled examples. Then opinion words are also refined.

Our main contributions are as follows:

- We propose a *Sentiment Graph Walking* algorithm to mine opinion words and opinion targets from reviews, which naturally incorporates confidence of syntactic pattern in a graph to improve extraction performance. To our best knowledge, the incorporation of pattern confidence in such a Sentiment Graph has never been studied before for opinion words/targets mining task (Section 3).
- We adopt a self-learning method for refining opinion words/targets generated by *Sentiment Graph Walking*. Specifically, it can remove high-frequency noise terms and capture long-tail opinion targets in corpora (Section 4).
- We perform experiments on three real world datasets, which demonstrate the effectiveness of our method compared with state-of-the-art unsupervised methods (Section 5).

## 2 Related Work

In opinion words/targets mining task, most unsupervised methods rely on identifying opinion relations between opinion words and opinion targets. Hu and Liu (2004) proposed an association mining technique to extract opinion words/targets. The simple heuristic rules they used may potentially introduce many false opinion words/targets. To identify opinion relations more precisely, subsequent research work exploited syntax information. Popescu and Etzioni (2005) used manually compiled syntactic patterns and Pointwise Mutual Information (PMI) to extract opinion words/targets. Qiu et al. (2009) proposed a bootstrapping framework called *Double Propagation* which introduced eight heuristic syntactic rules. While manually defining syntactic patterns could be time-consuming and error-prone, we learn syntactic patterns automatically from data.

There have been extensive works on mining opinion words and opinion targets by syntactic pattern learning. Riloff and Wiebe (2003) performed pattern learning through bootstrapping while extracting subjective expressions. Zhuang et al. (2006) obtained various dependency relationship templates from an annotated movie corpus and applied them to supervised opinion words/targets extraction. Kobayashi et al. (2007) adopted a supervised learning technique to search for useful syntactic patterns as contextual clues. Our approach is similar to (Wiebe and Riloff, 2005) and (Xu et al., 2013), all of which apply syntactic pattern learning and adopt self-learning strategy. However, the task of (Wiebe and Riloff, 2005) was to classify sentiment orientations in sentence level, while ours needs to extract more detailed information in term level. In addition, our method extends (Xu et al., 2013), and we give a more complete and in-depth analysis on the aforementioned problems in the first section. There were also many works employed graph-based method (Li et al., 2012; Zhang et al., 2010; Hassan and Radev, 2010; Liu et al., 2012), but none of previous works considered confidence of patterns in the graph.

In supervised approaches, various kinds of models were applied, such as HMM (Jin and Ho, 2009), SVM (Wu et al., 2009) and CRFs (Li et al., 2010). The downside of supervised methods was the difficulty of obtaining annotated training data in practical applications. Also, classifiers trained

on one domain often fail to give satisfactory results when shifted to another domain. Our method does not rely on annotated training data.

### 3 The First Stage: Sentiment Graph Walking Algorithm

In the first stage, we propose a graph-based algorithm called *Sentiment Graph Walking* to mine opinion words and opinion targets from reviews.

#### 3.1 Opinion Pattern Learning for Candidates Generation

For a given sentence, we first obtain its dependency tree. Following (Hu and Liu, 2004; Popescu and Etzioni, 2005; Qiu et al., 2009), we regard all adjectives as opinion word candidates (OC) and all nouns or noun phrases as opinion target candidates (TC). A statistic-based method in (Zhu et al., 2009) is used to detect noun phrases. Then candidates are replaced by wildcards “<OC>” or “<TC>”. Figure 1 gives a dependency tree example generated by Minipar (Lin, 1998).

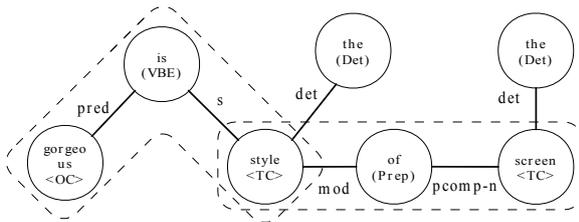


Figure 1: The dependency tree of the sentence “The style of the screen is gorgeous”.

We extract two kinds of opinion patterns: “OC-TC” pattern and “TC-TC” pattern. The “OC-TC” pattern is the shortest path between an OC wildcard and a TC wildcard in dependency tree, which captures opinion relation between an opinion word candidate and an opinion target candidate. Similarly, the “TC-TC” pattern captures opinion relation between two opinion target candidates.<sup>1</sup> Words in opinion patterns are replaced by their POS tags, and we constrain that there are at most two words other than wildcards in each pattern. In Figure 1, there are two opinion patterns marked out by dash lines: “<OC>{pred}{VBE}{s}<TC>” for the “OC-TC” type and “<TC>{mod}{(Prep)}{pcomp-n}<TC>” for the “TC-TC” type. After all pat-

<sup>1</sup>We do not identify the opinion relation “OC-OC” because this relation is often unreliable.

terns are generated, we drop those patterns with frequency lower than a threshold  $F$ .

#### 3.2 Sentiment Graph Construction

To model the opinion relations among opinion words/targets and opinion patterns, a graph named as *Sentiment Graph* is constructed, which is a weighted, directed graph  $G = (V, E, W)$ , where

- $V = \{V_{oc} \cup V_{tc} \cup V_p\}$  is the set of vertices in  $G$ , where  $V_{oc}$ ,  $V_{tc}$  and  $V_p$  represent the set of opinion word candidates, opinion target candidates and opinion patterns, respectively.
- $E = \{E_{po} \cup E_{pt}\} \subseteq \{V_p \times V_{oc}\} \cup \{V_p \times V_{tc}\}$  is the weighted, bi-directional edge set in  $G$ , where  $E_{po}$  and  $E_{pt}$  are mutually exclusive sets of edges connecting opinion word/target vertices to opinion pattern vertices. Note that there are no edges between  $V_{oc}$  and  $V_{tc}$ .
- $W : E \rightarrow \mathbb{R}^+$  is the weight function which assigns non-negative weight to each edge. For each  $(e : v_a \rightarrow v_b) \in E$ , where  $v_a, v_b \in V$ , the weight function  $w(v_a, v_b) = freq(v_a, v_b) / freq(v_a)$ , where  $freq(\cdot)$  is the frequency of a candidate extracted by opinion patterns or co-occurrence frequency between two candidates.

Figure 2 shows an example of Sentiment Graph.

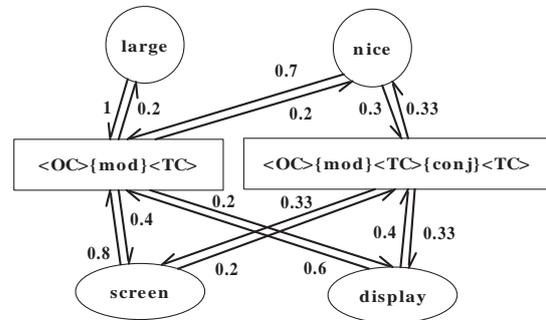


Figure 2: An example of Sentiment Graph.

#### 3.3 Confidence Estimation by Random Walking with Restart

We believe that considering confidence of patterns can potentially improve the extraction accuracy. Our intuitive idea is: (i) If an opinion word/target is with higher confidence, the syntactic patterns containing this term are more likely to be used to express customers’ opinion. (ii) If an opinion pattern has higher confidence, terms extracted by this pattern are more likely to be correct. It’s a reinforcement process.

We use Random Walking with Restart (RWR) algorithm to implement our idea described above. Let  $\mathbf{M}_{oc.p}$  denotes the transition matrix from  $V_{oc}$  to  $V_p$ , for  $v_o \in V_{oc}, v_p \in V_p$ ,  $\mathbf{M}_{oc.p}(v_o, v_p) = w(v_o, v_p)$ . Similarly, we have  $\mathbf{M}_{tc.p}$ ,  $\mathbf{M}_{p.oc}$ ,  $\mathbf{M}_{p.tc}$ . Let  $\mathbf{c}$  denotes confidence vector of candidates so  $\mathbf{c}_{oc}^t$ ,  $\mathbf{c}_{tc}^t$  and  $\mathbf{c}_p^t$  are confidence vectors for opinion word/target/pattern candidates after walking  $t$  steps. Initially  $\mathbf{c}_{oc}^0$  is uniformly distributed on a few domain-independent opinion word seeds, then the following formula are updated iteratively until  $\mathbf{c}_{tc}^t$  and  $\mathbf{c}_{oc}^t$  converge:

$$\mathbf{c}_p^{t+1} = \mathbf{M}_{oc.p}^T \times \mathbf{c}_{oc}^t + \mathbf{M}_{tc.p}^T \times \mathbf{c}_{tc}^t \quad (1)$$

$$\mathbf{c}_{oc}^{t+1} = (1 - \lambda)\mathbf{M}_{p.oc}^T \times \mathbf{c}_p^t + \lambda\mathbf{c}_{oc}^0 \quad (2)$$

$$\mathbf{c}_{tc}^{t+1} = \mathbf{M}_{p.tc}^T \times \mathbf{c}_p^t \quad (3)$$

where  $\mathbf{M}^T$  is the transpose of matrix  $\mathbf{M}$  and  $\lambda$  is a small probability of teleporting back to the seed vertices which prevents us from walking too far away from the seeds. In the experiments below,  $\lambda$  is set 0.1 empirically.

#### 4 The Second Stage: Refining Extracted Results Using Self-Learning

At the end of the first stage, we obtain a ranked list of opinion words and opinion targets, in which higher ranked terms are more likely to be correct. Nevertheless, there are still some issues needed to be addressed:

- 1) In the target candidate list, some high-frequency frivolous general nouns such as “thing” and “people” are also highly ranked. This is because there exist many opinion expressions containing non-target terms such as “good thing”, “nice people”, etc. in reviews. Due to the lack of ground-truth knowledge for opinion targets, the false opinion target problem still remains unsolved.
- 2) In another aspect, long-tail opinion targets may have low degree in Sentiment Graph. Hence their confidence will be low although they may be extracted by some high quality patterns. Therefore, the first stage is incapable of dealing with the long-tail opinion target problem.
- 3) Furthermore, the first stage also extracts some high-frequency false opinion words such as “every”, “many”, etc. Many terms of this kind are introduced by high-frequency false opinion targets, for there are large

amounts of phrases like “every time” and “many people”. So this issue is a side effect of the false opinion target problem.

To address these issues, we exploit a self-learning strategy. For opinion targets, we use a semi-supervised binary classifier called *target refining classifier* to refine target candidates. For opinion words, we use the classified list of opinion targets to further refine the extracted opinion word candidates.

#### 4.1 Opinion Targets Refinement

There are two keys for opinion target refinement: (i) How to generate the initial labeled data for target refining classifier. (ii) How to properly represent a long-tail opinion target candidate other than comparing frequency between different targets.

For the first key, it is clearly improper to select high-confidence targets as positive examples and choose low-confidence targets as negative examples<sup>2</sup>, for there are noise with high confidence and long-tail targets with low confidence. Fortunately, a large proportion of general noun noises are the most frequent words in common texts. Therefore, we can generate a small domain-independent general noun (GN) corpus from large web corpora to cover some most frequently used general noun examples. Then labeled examples can be drawn from the target candidate list and the GN corpus.

For the second key, we utilize opinion words and opinion patterns with their confidence scores to represent an opinion target. By this means, a long-tail opinion target can be determined by its own contexts, whose weights are learnt from contexts of frequent opinion targets. Thus, if a long-tail opinion target candidate has high contextual support, it will have higher probability to be found out in despite of its low frequency.

**Creation of General Noun Corpora.** 1000 most frequent nouns in Google-1-gram<sup>3</sup> were selected as general noun candidates. On the other hand, we added all nouns in the top three levels of hyponyms in four WordNet (Miller, 1995) synsets “object”, “person”, “group” and “measure” into the GN corpus. Our idea was based on the fact that a term is more general when it sits in higher level in the WordNet hierarchy. Then inapplicable candidates were discarded and a 3071-word English

<sup>2</sup>Note that the “positive” and “negative” here denote opinion targets and non-target terms respectively and they do not indicate sentiment polarities.

<sup>3</sup><http://books.google.com/ngrams>.

GN corpus was created. Another Chinese GN corpus with 3493 words was generated in the similar way from HowNet (Gan and Wong, 2000).

**Generation of Labeled Examples.** Let  $\mathcal{T} = \{\mathcal{Y}_{+1}, \mathcal{Y}_{-1}\}$  denotes the initial labeled set, where  $N$  most highly confident target candidates but not in our GN corpora are regarded as the positive example set  $\mathcal{Y}_{+1}$ , other  $N$  terms from GN corpora which are also top ranked in the target list are selected as the negative example set  $\mathcal{Y}_{-1}$ . The remainder unlabeled candidates are denoted by  $\mathcal{T}^*$ .

**Feature Representation for Classifier.** Given  $\mathcal{T}$  and  $\mathcal{T}^*$  in the form of  $\{(\mathbf{x}_i, y_i)\}$ . For a target candidate  $t_i$ ,  $\mathbf{x}_i = (o_1, \dots, o_n, p_1, \dots, p_m)^T$  represents its feature vector, where  $o_j$  is the opinion word feature and  $p_k$  is the opinion pattern feature. The value of feature is defined as follows,

$$x(o_j) = \text{conf}(o_j) \times \frac{\sum_{p_k} \text{freq}(t_i, o_j, p_k)}{\text{freq}(o_j)} \quad (4)$$

$$x(p_k) = \text{conf}(p_k) \times \frac{\sum_{o_j} \text{freq}(t_i, o_j, p_k)}{\text{freq}(p_k)} \quad (5)$$

where  $\text{conf}(\cdot)$  denotes confidence score estimated by RWR,  $\text{freq}(\cdot)$  has the same meaning as in Section 3.2. Particularly,  $\text{freq}(t_i, o_j, p_k)$  represents the frequency of pattern  $p_k$  extracting opinion target  $t_i$  and opinion word  $o_j$ .

**Target Refinement Classifier:** We use support vector machine as the binary classifier. Hence, the classification problem can be formulated as to find a hyperplane  $\langle \mathbf{w}, b \rangle$  that separates both labeled set  $\mathcal{T}$  and unlabeled set  $\mathcal{T}^*$  with maximum margin. The optimization goal is to minimize over  $(\mathcal{T}, \mathcal{T}^*, \mathbf{w}, b, \xi_1, \dots, \xi_n, \xi_1^*, \dots, \xi_k^*)$ :

$$\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=0}^n \xi_i + C^* \sum_{j=0}^k \xi_j^*$$

$$\text{subject to : } \forall_{i=1}^n : y_i [\mathbf{w} \cdot \mathbf{x}_i + b] \geq 1 - \xi_i$$

$$\forall_{j=1}^k : y_j^* [\mathbf{w} \cdot \mathbf{x}_j^* + b] \geq 1 - \xi_j^*$$

$$\forall_{i=1}^n : \xi_i > 0$$

$$\forall_{j=1}^k : \xi_j^* > 0$$

where  $y_i, y_j^* \in \{+1, -1\}$ ,  $\mathbf{x}_i$  and  $\mathbf{x}_j^*$  represent feature vectors,  $C$  and  $C^*$  are parameters set by user. This optimization problem can be implemented by a typical Transductive Support Vector Machine (TSVM) (Joachims, 1999).

## 4.2 Opinion Words Refinement

We use the classified opinion target results to refine opinion words by the following equation,

$$s(o_j) = \sum_{t_i \in T} \sum_{p_k} \frac{s(t_i) \text{conf}(p_k) \text{freq}(t_i, o_j, p_k)}{\text{freq}(t_i)}$$

where  $T$  is the opinion target set in which each element is classified as positive during opinion target refinement,  $s(t_i)$  denotes confidence score exported by the target refining classifier. Particularly,  $\text{freq}(t_i) = \sum_{o_j} \sum_{p_k} \text{freq}(t_i, o_j, p_k)$ . A higher score of  $s(o_j)$  means that candidate  $o_j$  is more likely to be an opinion word.

## 5 Experiments

### 5.1 Datasets and Evaluation Metrics

**Datasets:** We select three real world datasets to evaluate our approach. The first one is called *Customer Review Dataset (CRD)* (Hu and Liu, 2004) which contains reviews on five different products (represented by D1 to D5) in English. The second dataset is pre-annotated and published in *COAE08*<sup>4</sup>, where two domains of Chinese reviews are selected. At last, we employ a benchmark dataset in (Wang et al., 2011) and named it as *Large*. We manually annotated opinion words and opinion targets as the gold standard. Three annotators were involved. Firstly, two annotators were required to annotate out opinion words and opinion targets in sentences. When conflicts happened, the third annotator would make the final judgment. The average Kappa-values of the two domains were 0.71 for opinion words and 0.66 for opinion targets. Detailed information of our datasets is shown in Table 1.

Dataset	Domain	#Sentences	#OW	#OT
Large (English)	Hotel	10,000	434	1,015
	MP3	10,000	559	1,158
COAE08 (Chinese)	Camera	2,075	351	892
	Car	4,783	622	1,179

Table 1: The detailed information of datasets. OW stands for opinion words and OT stands for targets.

**Pre-processing:** Firstly, HTML tags are removed from texts. Then Minipar (Lin, 1998) is used to parse English corpora, and Stanford Parser (Chang et al., 2009) is used for Chinese

<sup>4</sup><http://ir-china.org.cn/coae2008.html>

Methods	D1			D2			D3			D4			D5			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	F
Hu	0.75	0.82	0.78	0.71	0.79	0.75	0.72	0.76	0.74	0.69	0.82	0.75	0.74	0.80	0.77	0.76
DP	<b>0.87</b>	0.81	<b>0.84</b>	<b>0.90</b>	0.81	<b>0.85</b>	<b>0.90</b>	0.86	<b>0.88</b>	0.81	0.84	0.82	<b>0.92</b>	0.86	<b>0.89</b>	<b>0.86</b>
Zhang	0.83	0.84	0.83	0.86	0.85	<b>0.85</b>	0.86	<b>0.88</b>	0.87	0.80	0.85	0.82	0.86	0.86	0.86	0.85
Ours-Stage1	0.79	<b>0.85</b>	0.82	0.82	<b>0.87</b>	0.84	0.83	0.87	0.85	0.78	<b>0.88</b>	0.83	0.82	<b>0.88</b>	0.85	0.84
Ours-Full	0.86	0.82	<b>0.84</b>	0.88	0.83	<b>0.85</b>	0.89	0.86	0.87	<b>0.83</b>	0.86	<b>0.84</b>	0.89	0.85	0.87	<b>0.86</b>

Table 2: Results of opinion target extraction on the *Customer Review Dataset*.

Methods	D1			D2			D3			D4			D5			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	F
Hu	0.57	<b>0.75</b>	0.65	0.51	0.76	0.61	0.57	0.73	0.64	0.54	0.62	0.58	0.62	0.67	0.64	0.62
DP	<b>0.64</b>	0.73	0.68	0.57	0.79	0.66	0.65	0.70	0.67	0.61	0.65	0.63	0.70	0.68	<b>0.69</b>	0.67
Ours-Stage1	0.61	<b>0.75</b>	0.67	0.55	<b>0.80</b>	0.65	0.63	<b>0.75</b>	<b>0.68</b>	0.60	<b>0.69</b>	0.64	0.68	<b>0.70</b>	<b>0.69</b>	0.67
Ours-Full	<b>0.64</b>	0.74	<b>0.69</b>	<b>0.59</b>	0.79	<b>0.68</b>	<b>0.66</b>	0.71	<b>0.68</b>	<b>0.65</b>	0.67	<b>0.66</b>	<b>0.72</b>	0.67	<b>0.69</b>	<b>0.68</b>

Table 3: Results of opinion word extraction on the *Customer Review Dataset*.

corpora. Stemming and fuzzy matching are also performed following previous work (Hu and Liu, 2004).

**Evaluation Metrics:** We evaluate our method by precision(P), recall(R) and F-measure(F).

## 5.2 Our Method vs. the State-of-the-art

Three state-of-the-art unsupervised methods are used as competitors to compare with our method.

*Hu* extracts opinion words/targets by using adjacency rules (Hu and Liu, 2004).

*DP* uses a bootstrapping algorithm named as *Double Propagation* (Qiu et al., 2009).

*Zhang* is an enhanced version of *DP* and employs HITS algorithm (Kleinberg, 1999) to rank opinion targets (Zhang et al., 2010).

*Ours-Full* is the full implementation of our method. We employ SVM<sup>light</sup> (Joachims, 1999) as the target refining classifier. Default parameters are used except the bias item is set 0.

*Ours-Stage1* only uses *Sentiment Graph Walking* algorithm which doesn't have opinion word and opinion target refinement.

All of the above approaches use same five common opinion word seeds. The choice of opinion seeds seems reasonable, as most people can easily come up with 5 opinion words such as "good", "bad", etc. The performance on five products of *CRD* dataset is shown in Table 2 and Table 3. *Zhang* does not extract opinion words so their results for opinion words are not taken into account. We can see that *Ours-Stage1* achieves superior recall but has some loss in precision compared with *DP* and *Zhang*. This may be because the *CRD* dataset is too small and our statistic-based method may suffer from data sparseness.

In spite of this, *Ours-Full* achieves comparable F-measure with *DP*, which is a well-designed rule-based method.

The results on two larger datasets are shown in Table 4 and Table 5, from which we can have the following observation: (i) All syntax-based-methods outperform *Hu*, showing the importance of syntactic information in opinion relation identification. (ii) *Ours-Full* outperforms the three competitors on all domains provided. (iii) *Ours-Stage1* outperforms *Zhang*, especially in terms of recall. We believe it benefits from our automatical pattern learning algorithm. Moreover, *Ours-Stage1* do not loss much in precision compared with *Zhang*, which indicates the applicability to estimate pattern confidence in Sentiment Graph. (iv) *Ours-Full* achieves 4-9% improvement in precision over the most accurate method, which shows the effectiveness of our second stage.

## 5.3 Detailed Discussions

This section gives several variants of our method to have a more detailed analysis.

*Ours-Bigraph* constructs a bi-graph between opinion words and targets, so opinion patterns are not included in the graph. Then RWR algorithm is used to only assign confidence to opinion word/target candidates.

*Ours-Stage2* only contains the second stage, which doesn't apply *Sentiment Graph Walking* algorithm. Hence the confidence score  $conf(\cdot)$  in Equations (4) and (5) have no values and they are set to 1. The initial labeled examples are exactly the same as *Ours-Full*. Due to the limitation of space, we only give analysis on opinion target extraction results in Figure 3.

Methods	MP3			Hotel			Camera			Car			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	F
Hu	0.53	0.55	0.54	0.55	0.57	0.56	0.63	0.65	0.64	0.62	0.58	0.60	0.58
DP	0.66	0.57	0.61	0.66	0.60	0.63	0.71	0.70	0.70	0.72	0.65	0.68	0.66
Zhang	0.65	0.62	0.63	0.64	0.66	0.65	0.71	0.78	0.74	0.69	0.68	0.68	0.68
Ours-Stage1	0.62	0.68	0.65	0.63	0.71	0.67	0.69	0.80	0.74	0.66	0.71	0.68	0.69
Ours-Full	<b>0.73</b>	<b>0.71</b>	<b>0.72</b>	<b>0.75</b>	<b>0.73</b>	<b>0.74</b>	<b>0.78</b>	<b>0.81</b>	<b>0.79</b>	<b>0.76</b>	<b>0.73</b>	<b>0.74</b>	<b>0.75</b>

Table 4: Results of opinion targets extraction on *Large* and *COAE08*.

Methods	MP3			Hotel			Camera			Car			Avg.
	P	R	F	P	R	F	P	R	F	P	R	F	F
Hu	0.48	0.65	0.55	0.51	0.68	0.58	0.72	0.74	0.73	0.70	0.71	0.70	0.64
DP	0.58	0.62	0.60	0.60	0.66	0.63	0.80	0.73	0.76	0.79	0.71	0.75	0.68
Ours-Stage1	0.59	<b>0.69</b>	0.64	0.61	<b>0.71</b>	0.66	0.79	<b>0.78</b>	0.78	0.77	<b>0.77</b>	0.77	0.71
Ours-Full	<b>0.64</b>	0.67	<b>0.65</b>	<b>0.67</b>	0.69	<b>0.68</b>	<b>0.82</b>	<b>0.78</b>	<b>0.80</b>	<b>0.80</b>	0.76	<b>0.78</b>	<b>0.73</b>

Table 5: Results of opinion words extraction on *Large* and *COAE08*.

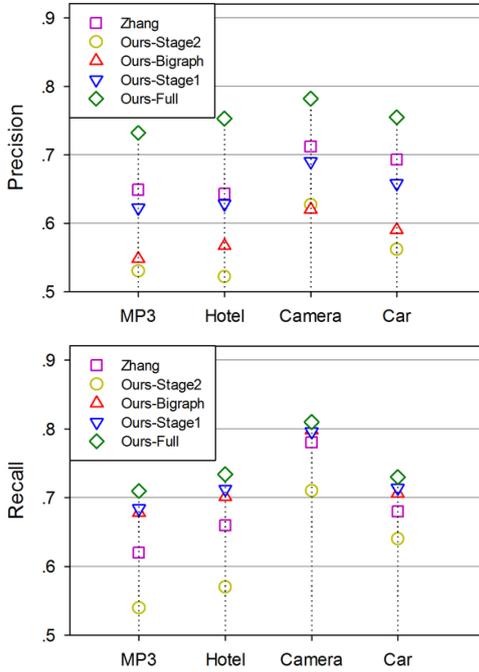


Figure 3: Opinion target extraction results.

### 5.3.1 The Effect of Sentiment Graph Walking

We can see that our graph-based methods (*Ours-Bigraph* and *Ours-Stage1*) achieve higher recall than *Zhang*. By learning patterns automatically, our method captures opinion relations more efficiently. Also, *Ours-Stage1* outperforms *Ours-Bigraph*, especially in precision. We believe it is because *Ours-Stage1* estimated confidence of patterns so false opinion relations are reduced. Therefore, the consideration of pattern confidence is beneficial as expected, which alleviates the false opinion relation problem. On another hand, we find that *Ours-Stage2* has much worse perfor-

mance than *Ours-Full*. This shows the effectiveness of *Sentiment Graph Walking* algorithm since the confidence scores estimated in the first stage are indispensable and indeed key to the learning of the second stage.

### 5.3.2 The Effect of Self-Learning

Figure 4 shows the average Precision@N curve of four domains on opinion target extraction. *Ours-GN-Only* is implemented by only removing 50 initial negative examples found by our GN corpora. We can see that the GN corpora work quite well, which find out most top-ranked false opinion targets. At the same time, *Ours-Full* has much better performance than *Ours-GN-Only* which indicates that *Ours-Full* can filter out more noises other than the initial negative examples. Therefore, our self-learning strategy alleviates the shortcoming of false opinion target problem. Moreover, Table 5 shows that the performance of opinion word extraction is also improved based on the classified results of opinion targets.

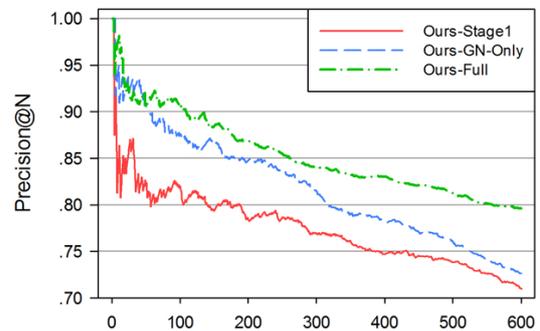


Figure 4: The average precision@N curve of the four domains on opinion target extraction.

ID	Pattern	Example	#Ext.	Conf.	PrO	PrT
#1	<OC>{mod}<TC>	it has a <b>clear</b> <i>screen</i>	7344	0.3938	0.59	0.66
#2	<TC>{subj}<OC>	the <i>sound quality</i> is <b>excellent</b>	2791	0.0689	0.62	0.70
#3	<TC>{conj}<TC>	the <i>size</i> and <i>weight</i> make it <b>convenient</b>	3620	0.0208	N/A	0.67
#4	<TC>{subj}<TC>	the <i>button layout</i> is a <b>simplistic</b> <i>plus</i>	1615	0.0096	N/A	0.67
#5	<OC>{pnmod}<TC>	the <i>buttons</i> <b>easier</b> to use	128	0.0014	0.61	0.34
#6	<TC>{subj}(V){s}(VBE){subj}<OC>	<i>software</i> provided is <b>simple</b>	189	0.0015	0.54	0.33
#7	<OC>{mod}(Prep){pcomp-c}(V){obj}<TC>	<b>great</b> for playing <i>audible books</i>	211	0.0013	0.43	0.48

Table 6: Examples of English patterns. #Ext. represent number of terms extracted, Conf. denotes confidence score estimated by RWR and PrO/PrT stand for precisions of extraction on opinion words/targets of a pattern respectively. Opinion words in examples are in bold and opinion targets are in italic.

Figure 5 gives the recall of long-tail opinion targets<sup>5</sup> extracted, where *Ours-Full* is shown to have much better performance than *Ours-Stage1* and the three competitors. This observation proves that our method can improve the limitation of long-tail opinion target problem.

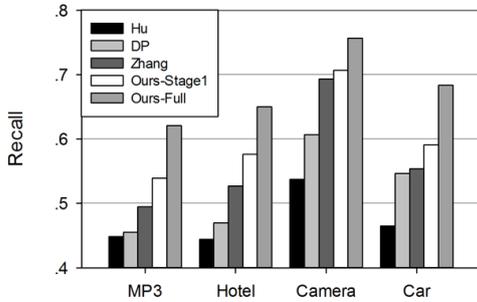


Figure 5: The recall of long-tail opinion targets.

### 5.3.3 Analysis on Opinion Patterns

Table 6 shows some examples of opinion pattern and their extraction accuracy on MP3 reviews in the first stage. Pattern #1 and #2 are the two most high-confidence opinion patterns of “OC-TC” type, and Pattern #3 and #4 demonstrate two typical “TC-TC” patterns. As these patterns extract too many terms, the overall precision is very low. We give Precision@400 of them, which is more meaningful because only top listed terms in the extracted results are regarded as opinion targets. Pattern #5 and #6 have high precision on opinion words but low precision on opinion targets. This observation demonstrates the false opinion target problem. Pattern #7 is a pattern example that extracts many false opinion relations and it has low precision for both opinion words and opinion targets. We can see that Pattern #7 has

<sup>5</sup>Since there is no explicit definition for the notion “long-tail”, we conservatively regard 60% opinion targets with the lowest frequency as the “long-tail” terms.

a lower confidence compared with Pattern #5 and #6 although it extracts more words. It’s because it has a low probability of walking from opinion seeds to this pattern. This further proves that our method can reduce the confidence of low-quality patterns.

### 5.3.4 Sensitivity of Parameters

Finally, we study the sensitivity of parameters when recall is fixed at 0.70. Figure 6 shows the precision curves at different  $N$  initial training examples and  $F$  filtering frequency. We can see that the performance saturates when  $N$  is set to 50 and it does not vary much under different  $F$ , showing the robustness of our method. We thus set  $N$  to 50, and  $F$  to 3 for *CRD*, 5 for *COAE08* and 10 for *Large* accordingly.

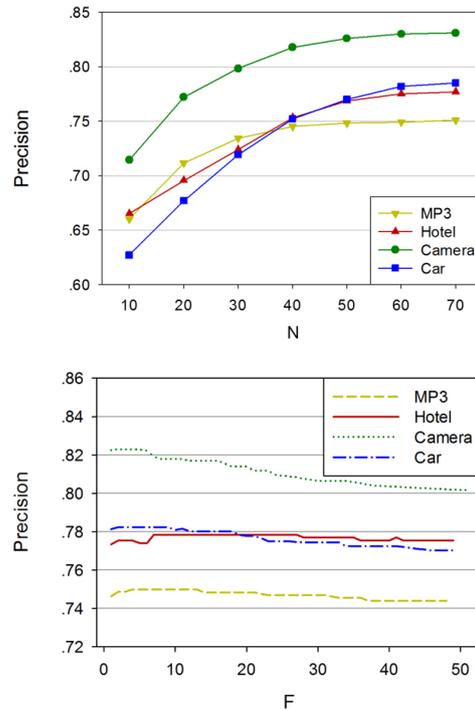


Figure 6: Influence of parameters.

## 6 Conclusion and Future Work

This paper proposes a novel two-stage framework for mining opinion words and opinion targets. In the first stage, we propose a *Sentiment Graph Walking* algorithm, which incorporates syntactic patterns in a Sentiment Graph to improve the extraction performance. In the second stage, we propose a self-learning method to refine the result of first stage. The experimental results show that our method achieves superior performance over state-of-the-art unsupervised methods.

We further notice that opinion words are not limited to adjectives but can also be other type of word such as verbs or nouns. Identifying all kinds of opinion words is a more challenging task. We plan to study this problem in our future work.

### Acknowledgement

Thanks to Prof. Yulan He for her insightful advices. This work was supported by the National Natural Science Foundation of China (No. 61070106, No. 61272332 and No. 61202329), the National High Technology Development 863 Program of China (No. 2012AA011102), the National Basic Research Program of China (No. 2012CB316300), Tsinghua National Laboratory for Information Science and Technology (TNList) Cross-discipline Foundation and the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (ICDD201201).

### References

- Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. Discriminative reordering with chinese grammatical relations features. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation, SSST '09*, pages 51–59.
- Sajib Dasgupta and Vincent Ng. 2009. Mine the easy, classify the hard: a semi-supervised approach to automatic sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2, ACL '09*, pages 701–709.
- Kok Wee Gan and Ping Wai Wong. 2000. Annotating information structures in chinese texts using hownet. In *Proceedings of the second workshop on Chinese language processing: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 12, CLPW '00*, pages 85–92, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ahmed Hassan and Dragomir Radev. 2010. Identifying text polarity using random walks. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 395–403, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '04*, pages 168–177, New York, NY, USA. ACM.
- Valentin Jijkoun, Maarten de Rijke, and Wouter Weerkamp. 2010. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pages 585–594, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Wei Jin and Hung Hay Ho. 2009. A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, pages 465–472.
- Thorsten Joachims. 1999. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209.
- Jason Kessler and Nicolas Nicolov. 2009. Targeting sentiment expressions through supervised ranking of linguistic configurations. In *Proceedings of the Third International AAAI Conference on Weblogs and Social Media*.
- Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September.
- Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1065–1074, June.
- Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING '10*, pages 653–661, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang, and Xiaoyan Zhu. 2012. Cross-domain co-extraction of sentiment and topic lexicons. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 410–419, July.

- Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Workshop on Evaluation of Parsing Systems at ICLRE*.
- Kang Liu, Liheng Xu, and Jun Zhao. 2012. Opinion target extraction using word-based translation model. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12*, pages 1346–1356, Stroudsburg, PA, USA. Association for Computational Linguistics.
- George A. Miller. 1995. Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41.
- Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05*, pages 339–346.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international joint conference on Artificial intelligence, IJCAI'09*, pages 1199–1204.
- Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing, EMNLP '03*, pages 105–112, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '11*, pages 618–626, New York, NY, USA. ACM.
- Janyce Wiebe and Ellen Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of the 6th international conference on Computational Linguistics and Intelligent Text Processing, CICLing'05*, pages 486–497.
- Yuanbin Wu, Qi Zhang, Xuanjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, pages 1533–1541.
- Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen, and Jun Zhao. 2013. Walk and learn: A two-stage approach for opinion words and opinion targets co-extraction. In *Proceedings of the 22nd International World Wide Web Conference, WWW '13*.
- Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1462–1470.
- Jingbo Zhu, Huizhen Wang, Benjamin K. Tsou, and Muhua Zhu. 2009. Multi-aspect opinion polling from textual reviews. In *Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09*, pages 1799–1802.
- Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management, CIKM '06*, pages 43–50.

# Connotation Lexicon: A Dash of Sentiment Beneath the Surface Meaning

Song Feng    Jun Seok Kang    Polina Kuznetsova    Yejin Choi

Department of Computer Science

Stony Brook University

Stony Brook, NY 11794-4400

songfeng, junkang, pkuznetsova, ychoi@cs.stonybrook.edu

## Abstract

Understanding the connotation of words plays an important role in interpreting subtle shades of sentiment beyond denotative or surface meaning of text, as seemingly objective statements often allude nuanced sentiment of the writer, and even purposefully conjure emotion from the readers' minds. The focus of this paper is drawing nuanced, connotative sentiments from even those words that are objective on the surface, such as “*intelligence*”, “*human*”, and “*cheesecake*”. We propose induction algorithms encoding a diverse set of linguistic insights (semantic prosody, distributional similarity, semantic parallelism of coordination) and prior knowledge drawn from lexical resources, resulting in the first *broad-coverage* connotation lexicon.

## 1 Introduction

There has been a substantial body of research in sentiment analysis over the last decade (Pang and Lee, 2008), where a considerable amount of work has focused on recognizing sentiment that is generally explicit and pronounced rather than implied and subdued. However in many real-world texts, even seemingly objective statements can be opinion-laden in that they often allude nuanced sentiment of the writer (Greene and Resnik, 2009), or purposefully conjure emotion from the readers' minds (Mohammad and Turney, 2010). Although some researchers have explored formal and statistical treatments of those implicit and implied sentiments (e.g. Wiebe et al. (2005), Esuli and Sebastiani (2006), Greene and Resnik (2009), Davidov et al. (2010)), automatic analysis of them largely remains as a big challenge.

In this paper, we concentrate on understanding the connotative sentiments of words, as they play an important role in interpreting subtle shades of sentiment beyond denotative or surface meaning of text. For instance, consider the following:

Geothermal replaces oil-heating; it helps reducing greenhouse *emissions*.<sup>1</sup>

Although this sentence could be considered as a factual statement from the general standpoint, the subtle effect of this sentence may not be entirely objective: this sentence is likely to have an influence on readers' minds in regard to their opinion toward “*geothermal*”. In order to sense the subtle overtone of sentiments, one needs to know that the word “*emissions*” has generally negative connotation, which geothermal *reduces*. In fact, depending on the pragmatic contexts, it could be precisely the intention of the author to transfer his opinion into the readers' minds.

The main contribution of this paper is a *broad-coverage* connotation lexicon that determines the connotative polarity of even those words with ever so subtle connotation beneath their surface meaning, such as “*Literature*”, “*Mediterranean*”, and “*wine*”. Although there has been a number of previous work that constructed sentiment lexicons (e.g., Esuli and Sebastiani (2006), Wilson et al. (2005a), Kaji and Kitsuregawa (2007), Qiu et al. (2009)), which seem to be increasingly and inevitably expanding over words with (strongly) connotative sentiments rather than explicit sentiments alone (e.g., “*gun*”), little prior work has directly tackled this problem of learning connotation,<sup>2</sup> and much of the subtle connotation of many seemingly objective words is yet to be determined.

<sup>1</sup>Our learned lexicon correctly assigns negative polarity to *emission*.

<sup>2</sup>A notable exception would be the work of Feng et al.

POSITIVE	NEGATIVE
FEMA, Mandela, Intel, Google, Python, Sony, Pulitzer, Harvard, Duke, Einstein, Shakespeare, Elizabeth, Clooney, Hoover, Goldman, Swarovski, Hawaii, Yellowstone	Katrina, Monsanto, Halliburton, Enron, Teflon, Hiroshima, Holocaust, Afghanistan, Mugabe, Hutu, Saddam, Osama, Qaeda, Kosovo, Helicobacter, HIV

Table 1: Example Named Entities (Proper Nouns) with Polar Connotation.

A central premise to our approach is that it is collocational statistics of words that affect and shape the polarity of connotation. Indeed, the etymology of “*connotation*” is from the Latin “*com-*” (“together or with”) and “*notare*” (“to mark”). It is important to clarify, however, that we do not simply assume that words that collocate share the same polarity of connotation. Although such an assumption played a key role in previous work for the analogous task of learning sentiment lexicon (Velikovich et al., 2010), we expect that the same assumption would be less reliable in drawing subtle connotative sentiments of words. As one example, the predicate “cure”, which has a positive connotation typically takes arguments with negative connotation, e.g., “disease”, when used as the “relieve” sense.<sup>3</sup>

Therefore, in order to attain a broad coverage lexicon while maintaining good precision, we guide the induction algorithm with multiple, carefully selected linguistic insights: [1] distributional similarity, [2] semantic parallelism of coordination, [3] selectional preference, and [4] semantic prosody (e.g., Sinclair (1991), Louw (1993), Stubbs (1995), Stefanowitsch and Gries (2003)), and also exploit existing lexical resources as an additional inductive bias.

We cast the connotation lexicon induction task as a collective inference problem, and consider approaches based on three distinct types of algorithmic framework that have been shown successful for conventional sentiment lexicon induction:

*Random walk based on HITS/PageRank* (e.g.,

Kleinberg (1999), Page et al. (1999), Feng et al. (2011) Heerschop et al. (2011), Montejo-Ráez et al. (2012))

*Label/Graph propagation* (e.g., Zhu and Ghahra-

(2011) but with practical limitations. See §3 for detailed discussion.

<sup>3</sup>Note that when “cure” is used as the “preserve” sense, it expects objects with non-negative connotation. Hence word-sense-disambiguation (WSD) presents a challenge, though not unexpectedly. In this work, we assume the general connotation of each word over statistically prevailing senses, leaving a more cautious handling of WSD as future work.

mani (2002), Velikovich et al. (2010))

*Constraint optimization* (e.g., Roth and Yih (2004), Choi and Cardie (2009), Lu et al. (2011)).

We provide comparative empirical results over several variants of these approaches with comprehensive evaluations including lexicon-based, human judgments, and extrinsic evaluations.

It is worthwhile to note that not all words have connotative meanings that are distinct from denotational meanings, and in some cases, it can be difficult to determine whether the overall sentiment is drawn from denotational or connotative meanings exclusively, or both. Therefore, we encompass any sentiment from either type of meanings into the lexicon, where non-neutral polarity prevails over neutral one if some meanings lead to neutral while others to non-neutral.<sup>4</sup>

Our work results in the first broad-coverage connotation lexicon,<sup>5</sup> significantly improving both the coverage and the precision of Feng et al. (2011). As an interesting by-product, our algorithm can be also used as a proxy to measure the general connotation of real-world named entities based on their collocational statistics. Table 1 highlights some example proper nouns included in the final lexicon.

The rest of the paper is structured as follows. In §2 we describe three types of induction algorithms followed by evaluation in §3. Then we revisit the induction algorithms based on constraint optimization in §4 to enhance quality and scalability. §5 presents comprehensive evaluation with human judges and extrinsic evaluations. Related work and conclusion are in §6 and §7.

<sup>4</sup>In general, polysemous words do not seem to have conflicting non-neutral polarities over different senses, though there are many exceptions, e.g., “heat”, or “fine”. We treat each word in each part-of-speech as a separate word to reduce such cases, otherwise aim to learn the most prevalent polarity in the corpus with respect to each part-of-speech of each word.

<sup>5</sup>Available at <http://www.cs.stonybrook.edu/~ychoi/connotation>.

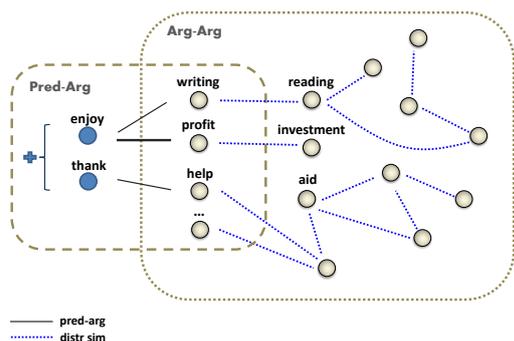


Figure 1: Graph for Graph Propagation (§2.2).

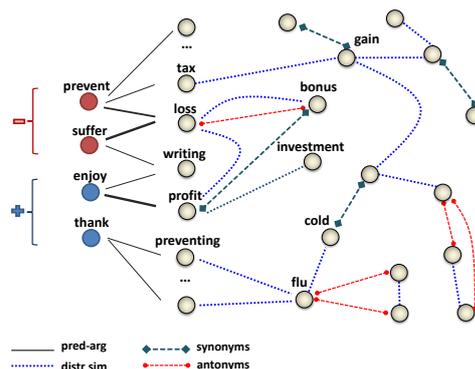


Figure 2: Graph for ILP/LP (§2.3, §4.2).

## 2 Connotation Induction Algorithms

We develop induction algorithms based on three distinct types of algorithmic framework that have been shown successful for the analogous task of sentiment lexicon induction: HITS & PageRank (§2.1), Label/Graph Propagation (§2.2), and Constraint Optimization via Integer Linear Programming (§2.3). As will be shown, each of these approaches will incorporate additional, more diverse linguistic insights.

### 2.1 HITS & PageRank

The work of Feng et al. (2011) explored the use of HITS (Kleinberg, 1999) and PageRank (Page et al., 1999) to induce the general connotation of words hinging on the linguistic phenomena of selectional preference and semantic prosody, i.e., *connotative predicates* influencing the connotation of their arguments. For example, the object of a negative connotative predicate “cure” is likely to have negative connotation, e.g., “disease” or “cancer”. The bipartite graph structure for this approach corresponds to the left-most box (labeled as “pred-arg”) in Figure 1.

### 2.2 Label Propagation

With the goal of obtaining a broad-coverage lexicon in mind, we find that relying only on the structure of semantic prosody is limiting, due to relatively small sets of connotative predicates available.<sup>6</sup> Therefore, we extend the graph structure as an *overlay of two sub-graphs* (Figure 1) as described below:

<sup>6</sup>For connotative predicates, we use the seed predicate set of Feng et al. (2011), which comprises of 20 positive and 20 negative predicates.

### Sub-graph #1: Predicate–Argument Graph

This sub-graph is the bipartite graph that encodes the selectional preference of connotative predicates over their arguments. In this graph, connotative predicates  $p$  reside on one side of the graph and their co-occurring arguments  $a$  reside on the other side of the graph based on Google Web 1T corpus.<sup>7</sup> The weight on the edges between the predicates  $p$  and arguments  $a$  are defined using Point-wise Mutual Information (PMI) as follows:

$$w(p \rightarrow a) := PMI(p, a) = \log_2 \frac{P(p, a)}{P(p)P(a)}$$

PMI scores have been widely used in previous studies to measure association between words (e.g., Turney (2001), Church and Hanks (1990)).

### Sub-graph #2: Argument–Argument Graph

The second sub-graph is based on the distributional similarities among the arguments. One possible way of constructing such a graph is simply connecting all nodes and assign edge weights proportionate to the word association scores, such as PMI, or distributional similarity. However, such a completely connected graph can be susceptible to propagating noise, and does not scale well over a very large set of vocabulary.

We therefore reduce the graph connectivity by exploiting *semantic parallelism of coordination* (Bock (1986), Hatzivassiloglou and McKeown

<sup>7</sup>We restrict predicate-argument pairs to verb-object pairs in this study. Note that Google Web 1T dataset consists of  $n$ -grams upto  $n = 5$ . Since  $n$ -gram sequences are too short to apply a parser, we extract verb-object pairs approximately by matching part-of-speech tags. Empirically, when overlaid with the second sub-graph, we found that it is better to keep the connectivity of this sub-graph as uni-directional. That is, we only allow edges to go from a predicate to an argument.

	POSITIVE	NEGATIVE	NEUTRAL
n.	avatar, adrenaline, keynote, debut, stakeholder, sunshine, cooperation	unbeliever, delay, shortfall, gunshot, misdemeanor, mutiny, rigor	header, mark, clothing, outline, grid, gasoline, course, preview
v.	handcraft, volunteer, party, accredit, personalize, nurse, google	sentence, cough, trap, scratch, debunk, rip, misspell, overcharge	state, edit, send, put, arrive, type, drill, name, stay, echo, register
a.	floral, vegetarian, prepared, ageless, funded, contemporary	debilitating, impaired, swollen, intentional, jarring, unearned	same, cerebral, west, uncut, automatic, hydrated, unheated, routine

Table 2: Example Words with Learned Connotation: Nouns(n), Verbs(v), Adjectives(a).

(1997), Pickering and Branigan (1998)). In particular, we consider an undirected edge between a pair of arguments  $a_1$  and  $a_2$  only if they occurred together in the “ $a_1$  and  $a_2$ ” or “ $a_2$  and  $a_1$ ” coordination, and assign edge weights as:

$$w(a_1 - a_2) = \text{CosineSim}(\vec{a}_1, \vec{a}_2) = \frac{\vec{a}_1 \cdot \vec{a}_2}{\|\vec{a}_1\| \|\vec{a}_2\|}$$

where  $\vec{a}_1$  and  $\vec{a}_2$  are co-occurrence vectors for  $a_1$  and  $a_2$  respectively. The co-occurrence vector for each word is computed using PMI scores with respect to the top  $n$  co-occurring words.<sup>8</sup>  $n$  (=50) is selected empirically. The edge weights in two sub-graphs are normalized so that they are in the comparable range.<sup>9</sup>

### Limitations of Graph-based Algorithms

Although graph-based algorithms (§2.1, §2.2) provide an intuitive framework to incorporate various lexical relations, limitations include:

1. They allow only *non-negative* edge weights. Therefore, we can encode only positive (supportive) relations among words (e.g., distributionally similar words will endorse each other with the same polarity), while missing on exploiting negative relations (e.g., antonyms may drive each other into the opposite polarity).
2. They induce positive and negative polarities in isolation via separate graphs. However, we expect that a more effective algorithm should induce both polarities simultaneously.
3. The framework does not readily allow incorporating a diverse set of *soft* and *hard* constraints.

<sup>8</sup>We discard edges with cosine similarity  $\leq 0$ , as those indicate either independence or the opposite of similarity.

<sup>9</sup>Note that cosine similarity does not make sense for the first sub-graph as there is no reason why a predicate and an argument should be distributionally similar. We experimented with many different variations on the graph structure and edge weights, including ones that include any word pairs that occurred frequently enough together. For brevity, we present the version that achieved the best results here.

### 2.3 Constraint Optimization

Addressing limitations of graph-based algorithms (§2.2), we propose an induction algorithm based on Integer Linear Programming (ILP). Figure 2 provides the pictorial overview. In comparison to Figure 1, two new components are: (1) dictionary-driven relations targeting enhanced *precision*, and (2) dictionary-driven words (i.e., unseen words with respect to those relations explored in Figure 1) targeting enhanced *coverage*. We formulate insights in Figure 2 using ILP as follows:

#### Definition of sets of words:

1.  $\mathcal{P}^+$ : the set of positive seed predicates.  
 $\mathcal{P}^-$ : the set of negative seed predicates.
2.  $\mathcal{S}$ : the set of seed sentiment words.
3.  $\mathcal{R}^{syn}$ : word pairs in synonyms relation.  
 $\mathcal{R}^{ant}$ : word pairs in antonyms relation.  
 $\mathcal{R}^{coord}$ : word pairs in coordination relation.  
 $\mathcal{R}^{pred}$ : word pairs in pred-arg relation.  
 $\mathcal{R}^{pred^{+(-)}}$ :  $\mathcal{R}^{pred}$  based on  $\mathcal{P}^+$  ( $\mathcal{P}^-$ ).

**Definition of variables:** For each word  $i$ , we define binary variables  $x_i, y_i, z_i \in \{0, 1\}$ , where  $x_i = 1$  ( $y_i = 1, z_i = 1$ ) if and only if  $i$  has a positive (negative, neutral) connotation respectively. For every pair of word  $i$  and  $j$ , we define binary variables  $d_{ij}^{pq}$  where  $p, q \in \{+, -, 0\}$  and  $d_{ij}^{pq} = 1$  if and only if the polarity of  $i$  and  $j$  are  $p$  and  $q$  respectively.

**Objective function:** We aim to maximize:

$$F = \Phi^{prosody} + \Phi^{coord} + \Phi^{neu}$$

where  $\Phi^{prosody}$  is the scores based on semantic prosody,  $\Phi^{coord}$  captures the distributional similarity over coordination, and  $\Phi^{neu}$  controls the sensitivity of connotation detection between positive (negative) and neutral. In particular,

$$\begin{aligned} \Phi^{prosody} &= \sum_{i,j}^{\mathcal{R}^{pred}} w_{i,j}^{pred} (d_{i,j}^{++} + d_{i,j}^{--} - d_{i,j}^{+-} - d_{i,j}^{-+}) \\ \Phi^{coord} &= \sum_{i,j}^{\mathcal{R}^{coord}} w_{i,j}^{coord} (d_{i,j}^{++} + d_{i,j}^{--} + d_{i,j}^{00}) \end{aligned}$$

$$\Phi^{neu} = \alpha \sum_{i,j}^{\mathcal{R}^{pred}} w_{i,j}^{pred} \cdot z_j$$

**Soft constraints (edge weights):** The weights in the objective function are set as follows:

$$w^{pred}(p, a) = \frac{freq(p, a)}{\sum_{(p,x) \in \mathcal{R}^{pred}} freq(p, x)}$$

$$w^{coord}(a_1, a_2) = CosSim(\vec{a}_1, \vec{a}_2) = \frac{\vec{a}_1 \cdot \vec{a}_2}{\|\vec{a}_1\| \|\vec{a}_2\|}$$

Note that the same  $w^{coord}(a_1, a_2)$  has been used in graph propagation described in Section 2.2.  $\alpha$  controls the sensitivity of connotation detection such that higher value of  $\alpha$  will promote neutral connotation over polar ones.

**Hard constrains for variable consistency:**

1. Each word  $i$  has one of  $\{+, -, \emptyset\}$  as polarity:  
 $\forall i, x_i + y_i + z_i = 1$
2. Variable consistency between  $d_{ij}^{pq}$  and  $x_i, y_i, z_i$ :

$$x_i + x_j - 1 \leq 2d_{i,j}^{++} \leq x_i + x_j$$

$$y_i + y_j - 1 \leq 2d_{i,j}^{--} \leq y_i + y_j$$

$$z_i + z_j - 1 \leq 2d_{i,j}^{00} \leq z_i + z_j$$

$$x_i + y_j - 1 \leq 2d_{i,j}^{+-} \leq x_i + y_j$$

$$y_i + x_j - 1 \leq 2d_{i,j}^{-+} \leq y_i + x_j$$

**Hard constrains for WordNet relations:**

1.  $\mathcal{C}^{ant}$ : Antonym pairs will not have the same positive or negative polarity:

$$\forall (i, j) \in \mathcal{R}^{ant}, x_i + x_j \leq 1, y_i + y_j \leq 1$$

For this constraint, we only consider antonym pairs that share the same root, e.g., “sufficient” and “insufficient”, as those pairs are more likely to have the opposite polarities than pairs without sharing the same root, e.g., “east” and “west”.

2.  $\mathcal{C}^{syn}$ : Synonym pairs will not have the opposite polarity:

$$\forall (i, j) \in \mathcal{R}^{syn}, x_i + y_j \leq 1, x_j + y_i \leq 1$$

### 3 Experimental Result I

We provide comprehensive comparisons over variants of three types of algorithms proposed in §2. We use the Google Web 1T data (Brants and Franz (2006)), and POS-tagged ngrams using Stanford POS Tagger (Toutanova and Manning (2000)). We filter out the ngrams with punctuations and other special characters to reduce the noise.

### 3.1 Comparison against Conventional Sentiment Lexicon

Note that we consider the connotation lexicon to be inclusive of a sentiment lexicon for two practical reasons: first, it is highly unlikely that any word with non-neutral sentiment (i.e., positive or negative) would carry connotation of the opposite, i.e., conflicting<sup>10</sup> polarity. Second, for some words with distinct sentiment or strong connotation, it can be difficult or even unnatural to draw a precise distinction between connotation and sentiment, e.g., “efficient”. Therefore, sentiment lexicons can serve as a surrogate to measure a subset of connotation words induced by the algorithms, as shown in Table 3 with respect to General Inquirer (Stone and Hunt (1963)) and MPQA (Wilson et al. (2005b)).<sup>11</sup>

**Discussion** Table 3 shows the agreement statistics with respect to two conventional sentiment lexicons. We find that the use of label propagation alone [PRED-ARG (CP)] improves the performance substantially over the comparable graph construction with different graph analysis algorithms, in particular, HITS and PageRank approaches of Feng et al. (2011). The two completely connected variants of the graph propagation on the Pred-Arg graph, [ $\otimes$  PRED-ARG (PMI)] and [ $\otimes$  PRED-ARG (CP)], do not necessarily improve the performance over the simpler and computationally lighter alternative, [PRED-ARG (CP)]. The [OVERLAY], which is based on both Pred-Arg and Arg-Arg subgraphs (§2.2), achieves the best performance among graph-based algorithms, significantly improving the precision over all other baselines. This result suggests:

- 1 The sub-graph #2, based on the semantic parallelism of coordination, is simple and yet very powerful as an inductive bias.
- 2 The performance of graph propagation varies significantly depending on the graph topology and the corresponding edge weights.

Note that a direct comparison against ILP for top N words is tricky, as ILP does not *rank* results. Only for comparison purposes however, we assign

<sup>10</sup>We consider “positive” and “negative” polarities conflict, but “neutral” polarity does *not* conflict with any.

<sup>11</sup>In the case of General Inquirer, we use words in POSITIV and NEGATIV sets as words with positive and negative labels respectively.

	GENINQ EVAL					MPQA EVAL				
	100	1,000	5,000	10,000	ALL	100	1,000	5,000	10,000	ALL
ILP	<b>97.6</b>	94.5	<b>84.5</b>	80.8	80.4	<b>98.0</b>	89.7	<b>84.6</b>	<b>81.2</b>	<b>78.4</b>
OVERLAY	97.0	<b>95.1</b>	78.8	(78.3)	78.3	<b>98.0</b>	<b>93.4</b>	82.1	77.7	77.7
⊗ PRED-ARG (PMI)	91.0	91.4	76.1	(76.1)	76.1	88.0	89.1	78.8	75.1	75.1
⊗ PRED-ARG (CP)	88.0	85.4	76.2	(76.2)	76.2	87.0	82.6	78.0	76.3	76.3
PRED-ARG (CP)	91.0	91.0	81.0	<b>(81.0)</b>	<b>81.0</b>	88.0	91.5	80.0	78.3	78.3
HITS-ASYMT	77.0	68.8	-	-	66.5	86.3	81.3	-	-	72.2
PAGERANK-ASYMF	77.0	68.5	-	-	65.7	87.2	80.3	-	-	72.3

Table 3: Evaluation of Induction Algorithms (§2) with respect to Sentiment Lexicons (precision%).

ranks based on the frequency of words for ILP. Because of this issue, the performance of top  $\sim 1k$  words of ILP should be considered only as a conservative measure. Importantly, when evaluated over more than top 5k words, ILP is overall the top performer considering both precision (shown in Table 3) and coverage (omitted for brevity).<sup>12</sup>

#### 4 Precision, Coverage, and Efficiency

In this section, we address three important aspects of an ideal induction algorithm: *precision*, *coverage*, and *efficiency*. For brevity, the remainder of the paper will focus on the algorithms based on constraint optimization, as it turned out to be the most effective one from the empirical results in §3.

**Precision** In order to see the effectiveness of the induction algorithms more sharply, we had used a limited set of seed words in §3. However to build a lexicon with substantially enhanced precision, we will use as a large seed set as possible, e.g., entire sentiment lexicons<sup>13</sup>.

**Broad coverage** Although statistics in Google 1T corpus represent a very large amount of text, words that appear in pred-arg and coordination relations are still limited. To substantially increase the coverage, we will leverage dictionary words (that are not in the corpus) as described in §2.3 and Figure 2.

**Efficiency** One practical problem with ILP is efficiency and scalability. In particular, we found that it becomes nearly impractical to run the ILP formulation including all words in WordNet plus all words in the argument position in Google Web 1T. We therefore explore an alternative approach based on Linear Programming in what follows.

<sup>12</sup>In fact, the performance of PRED-ARG variants for top 10K w.r.t. GENINQ is not meaningful as no additional word was matched beyond top 5k words.

<sup>13</sup>Note that doing so will prevent us from evaluating against the same sentiment lexicon used as a seed set.

#### 4.1 Induction using Linear Programming

One straightforward option for Linear Programming formulation may seem like using the same Integer Linear Programming formulation introduced in §2.3, only changing the variable definitions to be real values  $\in [0, 1]$  rather than integers. However, because the hard constraints in §2.3 are defined based on the assumption that all the variables are binary integers, those constraints are not as meaningful when considered for real numbers. Therefore we revise those hard constraints to encode various semantic relations (WordNet and semantic coordination) more directly.

**Definition of variables:** For each word  $i$ , we define variables  $x_i, y_i, z_i \in [0, 1]$ .  $i$  has a positive (negative) connotation if and only if the  $x_i$  ( $y_i$ ) is assigned the greatest value among the three variables; otherwise,  $i$  is neutral.

**Objective function:** We aim to maximize:

$$\begin{aligned}
 F &= \Phi^{prosody} + \Phi^{coord} + \Phi^{syn} + \Phi^{ant} + \Phi^{neu} \\
 \Phi^{prosody} &= \sum_{i,j}^{\mathcal{R}^{pred^+}} w_{i,j}^{pred^+} \cdot x_j + \sum_{i,j}^{\mathcal{R}^{pred^-}} w_{i,j}^{pred^-} \cdot y_j \\
 \Phi^{coord} &= \sum_{i,j}^{\mathcal{R}^{coord}} w_{i,j}^{coord} \cdot (dc_{i,j}^{++} + dc_{i,j}^{--}) \\
 \Phi^{syn} &= W^{syn} \sum_{i,j}^{\mathcal{R}^{syn}} (ds_{i,j}^{++} + ds_{i,j}^{--}) \\
 \Phi^{ant} &= W^{ant} \sum_{i,j}^{\mathcal{R}^{ant}} (da_{i,j}^{++} + da_{i,j}^{--}) \\
 \Phi^{neu} &= \alpha \sum_{i,j}^{\mathcal{R}^{pred}} w_{i,j}^{pred} \cdot z_j
 \end{aligned}$$

**Hard constraints** We add penalties to the objective function if the polarity of a pair of words is not consistent with its corresponding semantic relations. For example, for synonyms  $i$  and  $j$ , we introduce a penalty  $W^{syn}$  (a positive constant) for  $ds_{i,j}^{++}, ds_{i,j}^{--} \in [-1, 0]$ , where we set the upper bound of  $ds_{i,j}^{++}$  ( $ds_{i,j}^{--}$ ) as the signed distance of

	FORMULA	POSITIVE			NEGATIVE			ALL		
		R	P	F	R	P	F	R	P	F
ILP	$\Phi^{prosody} + \mathcal{C}^{syn} + \mathcal{C}^{ant}$	51.4	85.7	64.3	44.7	87.9	59.3	48.0	86.8	61.8
	$\Phi^{prosody} + \mathcal{C}^{syn} + \mathcal{C}^{ant} + \mathcal{C}^S$	61.2	93.3	73.9	52.4	<b>92.2</b>	66.8	56.8	<b>92.8</b>	70.5
	$\Phi^{prosody} + \Phi^{coord} + \mathcal{C}^{syn} + \mathcal{C}^{ant}$	67.3	75.0	70.9	53.7	84.4	65.6	60.5	79.7	68.8
	$\Phi^{prosody} + \Phi^{coord} + \mathcal{C}^{syn} + \mathcal{C}^{ant} + \mathcal{C}^S$	62.2	<b>96.0</b>	75.5	51.5	89.5	65.4	56.9	<b>92.8</b>	70.5
LP	$\Phi^{prosody} + \Phi^{syn} + \Phi^{ant}$	24.4	76.0	36.9	23.6	78.8	36.3	24.0	77.4	36.6
	$\Phi^{prosody} + \Phi^{syn} + \Phi^{ant} + \Phi^S$	71.6	87.8	78.9	68.8	84.6	75.9	70.2	86.2	77.4
	$\Phi^{prosody} + \Phi^{coord} + \Phi^{syn} + \Phi^{ant}$	67.9	92.6	78.3	64.6	89.1	74.9	66.3	90.8	76.6
	$\Phi^{prosody} + \Phi^{coord} + \Phi^{syn} + \Phi^{ant} + \Phi^S$	<b>78.6</b>	90.5	<b>84.1</b>	<b>73.3</b>	87.1	<b>79.6</b>	<b>75.9</b>	88.8	<b>81.8</b>

Table 4: ILP/LP Comparison on MQPA' (%).

$x_i$  and  $x_j$  ( $y_i$  and  $y_j$ ) as shown below:

For  $(i, j) \in \mathcal{R}^{syn}$ ,

$$ds_{i,j}^{++} \leq x_i - x_j, \quad ds_{i,j}^{++} \leq x_j - x_i$$

$$ds_{i,j}^{--} \leq y_i - y_j, \quad ds_{i,j}^{--} \leq y_j - y_i$$

Notice that  $ds_{i,j}^{++}, ds_{i,j}^{--}$  satisfying above inequalities will be always of negative values, hence in order to maximize the objective function, the LP solver will try to minimize the absolute values of  $ds_{i,j}^{++}, ds_{i,j}^{--}$ , effectively pushing  $i$  and  $j$  toward the same polarity. Constraints for semantic coordination  $\mathcal{R}^{coord}$  can be defined similarly. Lastly, following constraints encode antonym relations:

For  $(i, j) \in \mathcal{R}^{ant}$ ,

$$da_{i,j}^{++} \leq x_i - (1 - x_j), \quad da_{i,j}^{++} \leq (1 - x_j) - x_i$$

$$da_{i,j}^{--} \leq y_i - (1 - y_j), \quad da_{i,j}^{--} \leq (1 - y_j) - y_i$$

**Interpretation** Unlike ILP, some of the variables result in fractional values. We consider a word has positive or negative polarity only if the assignment indicates 1 for the corresponding polarity and 0 for the rest. In other words, we treat all words with fractional assignments over different polarities as neutral. Because the optimal solutions of LP correspond to extreme points in the convex polytope formed by the constraints, we obtain a large portion of words with non-fractional assignments toward non-neutral polarities. Alternatively, one can round up fractional values.

## 4.2 Empirical Comparisons: ILP v.s. LP

To solve the ILP/LP, we run ILOG CPLEX Optimizer (CPLEX, 2009) on a 3.5GHz 6 core CPU machine with 96GB RAM. Efficiency-wise, LP runs within 10 minutes while ILP takes several hours. Table 4 shows the results evaluated against MPQA for different variations of ILP and LP. We find that LP variants much better recall and F-score, while maintaining comparable precision.

Therefore, we choose the connotation lexicon by LP (C-LP) in the following evaluations in §5.

## 5 Experimental Results II

In this section, we present comprehensive intrinsic §5.1 and extrinsic §5.2 evaluations comparing three representative lexicons from §2 & §4: C-LP, OVERLAY, PRED-ARG (CP), and two popular sentiment lexicons: SentiWordNet (Baccianella et al., 2010) and GI+MPQA.<sup>14</sup> Note that C-LP is the largest among all connotation lexicons, including  $\sim 70,000$  polar words.<sup>15</sup>

### 5.1 Intrinsic Evaluation: Human Judgements

We evaluate 4000 words<sup>16</sup> using Amazon Mechanical Turk (AMT). Because we expect that judging a connotation can be dependent on one's cultural background, personality and value systems, we gather judgements from 5 people for each word, from which we hope to draw a more general judgement of connotative polarity. About 300 unique Turkers participated the evaluation tasks. We gather gold standard only for those words for which more than half of the judges agreed on the same polarity. Otherwise we treat them as ambiguous cases.<sup>17</sup> Figure 3 shows a part of the AMT task, where Turkers are presented with questions that help judges to determine the subtle connotative polarity of each word, then asked to rate the degree of connotation on a scale from -5 (most negative) and 5 (most positive). To draw

<sup>14</sup>GI+MPQA is the union of General Inquirer and MPQA. The GI, we use words in the "Positiv" & "Negativ" set. For SentiWordNet, to retrieve the polarity of a given word, we sum over the polarity scores over all senses, where positive (negative) values correspond to positive (negative) polarity.

<sup>15</sup> $\sim 13k$  adj,  $\sim 6k$  verbs,  $\sim 28k$  nouns,  $\sim 22k$  proper nouns.

<sup>16</sup>We choose words that are not already in GI+MPQA and obtain most frequent 10,000 words based on the unigram frequency in Google-Ngram, then randomly select 4000 words.

<sup>17</sup>We allow Turkers to mark words that can be used with both positive and negative connotation, which results in about 7% of words that are excluded from the gold standard set.

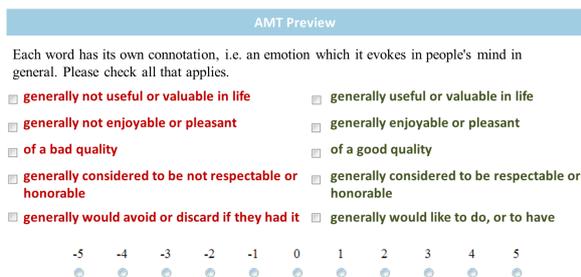


Figure 3: A Part of AMT Task Design.

QUESTION	YES		NO	
	%	Avg	%	Avg
“Enjoyable or pleasant”	43.3	2.9	16.3	-2.4
“Of a good quality”	56.7	2.5	6.1	-2.7
“Respectable / honourable”	21.0	3.3	14.0	-1.1
“Would like to do or have”	52.5	2.8	11.5	-2.4

Table 5: Distribution of Answers from AMT.

the gold standard, we consider two different voting schemes:

- $\Omega^{Vote}$ : The judgement of each Turker is mapped to neutral for  $-1 \leq \text{score} \leq 1$ , positive for  $\text{score} \geq 2$ , negative for  $\text{score} \leq -2$ , then we take the majority vote.
- $\Omega^{Score}$ : Let  $\sigma(i)$  be the sum (weighted vote) of the scores given by 5 judges for word  $i$ . Then we determine the polarity label  $l(i)$  of  $i$  as:

$$l(i) = \begin{cases} \text{positive} & \text{if } \sigma(i) > 1 \\ \text{negative} & \text{if } \sigma(i) < -1 \\ \text{neutral} & \text{if } -1 \leq \sigma(i) \leq 1 \end{cases}$$

The resulting distribution of judgements is shown in Table 5 & 6. Interestingly, we observe that *among the relatively frequently used English words, there are overwhelmingly more positively connotative words than negative ones.*

In Table 7, we show the percentage of words with the same label over the mutual words by the two lexicon. The highest agreement is 77% by C-LP and the gold standard by  $\text{AMT}^{Vote}$ . How good is this? It depends on what is the natural degree of agreement over subtle connotation among people. Therefore, we also report the degree of agreement among human judges in Table 7, where we compute the agreement of one Turker with respect to the gold standard drawn from the rest of the Turkers, and take the average across over all five Turkers<sup>18</sup>. Interestingly, the performance of

<sup>18</sup>In order to draw the gold standard from the 4 remaining Turkers, we consider adjusted versions of  $\Omega^{Vote}$  and  $\Omega^{Score}$  schemes described above.

	POS	NEG	NEU	UNDETERMINED
$\Omega^{Vote}$	50.4	14.6	24.1	10.9
$\Omega^{Score}$	67.9	20.6	11.5	n/a

Table 6: Distribution of Connotative Polarity from AMT.

	C-LP	SENTIWN	HUMAN JUDGES
$\Omega^{Vote}$	77.0	71.5	66.0
$\Omega^{Score}$	73.0	69.0	69.0

Table 7: Agreement (Accuracy) against AMT-driven Gold Standard.

Turkers is not as good as that of C-LP lexicon. We conjecture that this could be due to generally varying perception of different people on the connotative polarity,<sup>19</sup> while the corpus-driven induction algorithms focus on the *general* connotative polarity corresponding to the most prevalent senses of words in the corpus.

## 5.2 Extrinsic Evaluation

We conduct lexicon-based binary sentiment classification on the following two corpora.

**SemEval** From the SemEval task, we obtain a set of news headlines with annotated scores (ranging from -100 to 87). The positive/negative scores indicate the degree of positive/negative polarity orientation. We construct several sets of the positive and negative texts by setting thresholds on the scores as shown in Table 8. “ $\leq n$ ” indicates that the positive set consists of the texts with scores  $\geq n$  and the negative set consists of the texts with scores  $\leq -n$ .

**Emoticon tweets** The sentiment Twitter data<sup>20</sup> consists of tweets containing either a smiley emoticon (positive sentiment) or a frowny emoticon (negative sentiment). We filter out the tweets with question marks or more than 30 words, and keep the ones with at least two words in the union of all polar words in the five lexicons in Table 8, and then randomly select 10000 per class.

We denote the short text (e.g., content of tweets or headline texts from SemEval) by  $t$ .  $w$  represents the word in  $t$ .  $W^+/W^-$  is the set of posi-

<sup>19</sup>Pearson correlation coefficient among turkers is 0.28, which corresponds to a positive small to medium correlation. Note that when the annotation of turkers is aggregated, we observe agreement as high as 77% with respect to the learned connotation lexicon.

<sup>20</sup><http://www.stanford.edu/~alecmgo/cs224n/twitterdata.2009.05.25.c.zip>

LEXICON	DATA				
	TWEET	SEMEVAL			
		≤20	≤40	≤60	≤80
C-LP	<b>70.1</b>	<b>70.8</b>	<b>74.6</b>	<b>80.8</b>	<b>93.5</b>
OVERLAY	68.5	70.0	72.9	76.8	89.6
PRED-ARG (CP)	60.5	64.2	69.3	70.3	79.2
SENTIWN	67.4	61.0	64.5	70.5	79.0
GI+MPQA	65.0	64.5	69.0	74.0	80.5

Table 8: Accuracy on Sentiment Classification (%).

tive/negative words of the lexicon. We define the weight of  $w$  as  $s(w)$ . If  $w$  is adjective,  $s(w) = 2$ ; otherwise  $s(w) = 1$ . Then the polarity of each text is determined as follows:

$$pol(t) = \begin{cases} positive & \text{if } \sum_{w \in t}^{W^+} s(w) \geq \sum_{w \in t}^{W^-} s(w) \\ negative & \text{if } \sum_{w \in t}^{W^+} s(w) < \sum_{w \in t}^{W^-} s(w) \end{cases}$$

As shown in Table 8, C-LP generally performs better than the other lexicons on both corpora. Considering that only very simple classification strategy is applied, the result by the connotation lexicon is quite promising.

Finally, Table 1 highlights interesting examples of proper nouns with connotative polarity, e.g., “Mandela”, “Google”, “Hawaii” with positive connotation, and “Monsanto”, “Halliburton”, “Enron” with negative connotation, suggesting that our algorithms could potentially serve as a proxy to track the general connotation of real world entities. Table 2 shows example common nouns with connotative polarity.

### 5.3 Practical Remarks on WSD and MWEs

In this work we aim to find the polarity of most prevalent senses of each word, in part because it is not easy to perform unsupervised word sense disambiguation (WSD) on a large corpus in a reliable way, especially when the corpus consists primarily of short  $n$ -grams. Although the resulting lexicon loses on some of the polysemous words with potentially opposite polarities, per-word connotation (rather than per-sense connotation) does have a practical value: it provides a convenient option for users who wish to avoid the burden of WSD before utilizing the lexicon. Future work includes handling of WSD and multi-word expressions (MWEs), e.g., “Great Leader” (for Kim Jong-Il), “Inglourious Basterds” (a movie title).<sup>21</sup>

<sup>21</sup>These examples credit to an anonymous reviewer.

## 6 Related Work

A very interesting work of Mohammad and Turney (2010) uses Mechanical Turk in order to build the lexicon of emotions evoked by words. In contrast, we present an automatic approach that infers the general connotation of words. Velikovich et al. (2010) use graph propagation algorithms for constructing a web-scale polarity lexicon for sentiment analysis. Although we employ the same graph propagation algorithm, our graph construction is fundamentally different in that we integrate stronger inductive biases into the graph topology and the corresponding edge weights. As shown in our experimental results, we find that judicious construction of graph structure, exploiting multiple complementing linguistic phenomena can enhance both the performance and the efficiency of the algorithm substantially. Other interesting approaches include one based on min-cut (Dong et al., 2012) or LDA (Xie and Li, 2012). Our proposed approaches are more suitable for encoding a much diverse set of linguistic phenomena however. But our work use a few seed predicates with selectional preference instead of relying on word similarity. Some recent work explored the use of constraint optimization framework for inducing domain-dependent sentiment lexicon (Choi and Cardie (2009), Lu et al. (2011)). Our work differs in that we provide comprehensive insights into different formulations of ILP and LP, aiming to learn the much different task of learning the general connotation of words.

## 7 Conclusion

We presented a broad-coverage connotation lexicon that determines the subtle nuanced sentiment of even those words that are objective on the surface, including the general connotation of real-world named entities. Via a comprehensive evaluation, we provided empirical insights into three different types of induction algorithms, and proposed one with good precision, coverage, and efficiency.

## Acknowledgments

This research was supported in part by the Stony Brook University Office of the Vice President for Research. We thank reviewers for many insightful comments and suggestions, and for providing us with several very inspiring examples to work with.

## References

- Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).
- J. Kathryn Bock. 1986. Syntactic persistence in language production. *Cognitive psychology*, 18(3):355–387.
- Thorsten Brants and Alex Franz. 2006. {Web 1T 5-gram Version 1}.
- Yejin Choi and Claire Cardie. 2009. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 590–598, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Comput. Linguist.*, 16:22–29, March.
- ILOG CPLEX. 2009. High-performance software for mathematical programming and optimization. *URL* <http://www.ilog.com/products/cplex>.
- Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 107–116, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Xishuang Dong, Qibo Zou, and Yi Guan. 2012. Set-similarity joins based semi-supervised sentiment analysis. In *Neural Information Processing*, pages 176–183. Springer.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06)*, pages 417–422.
- Song Feng, Ritwik Bose, and Yejin Choi. 2011. Learning general connotation of words using graph-based algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1092–1103. Association for Computational Linguistics.
- Stephan Greene and Philip Resnik. 2009. More than words: Syntactic packaging and implicit sentiment. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 503–511, Boulder, Colorado, June. Association for Computational Linguistics.
- Vasileios Hatzivassiloglou and Kathleen R McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 174–181. Association for Computational Linguistics.
- Bas Heerschop, Alexander Hogenboom, and Flavius Frasincar. 2011. Sentiment lexicon creation from lexical resources. In *Business Information Systems*, pages 185–196. Springer.
- Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building lexicon for sentiment analysis from massive collection of html documents. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.
- Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *JOURNAL OF THE ACM*, 46(5):604–632.
- Bill Louw. 1993. Irony in the text or insincerity in the writer. *Text and technology: In honour of John Sinclair*, pages 157–176.
- Yue Lu, Malu Castellanos, Umeshwar Dayal, and ChengXiang Zhai. 2011. Automatic construction of a context-aware sentiment lexicon: an optimization approach. In *Proceedings of the 20th international conference on World wide web*, pages 347–356. ACM.
- Saif Mohammad and Peter Turney. 2010. Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 26–34, Los Angeles, CA, June. Association for Computational Linguistics.
- Arturo Montejo-Ráez, Eugenio Martínez-Cámara, M. Teresa Martín-Valdivia, and L. Alfonso Ureña López. 2012. Random walk weighting over sentiwordnet for sentiment polarity detection on twitter. In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, pages 3–10, Jeju, Korea, July. Association for Computational Linguistics.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.
- Martin J Pickering and Holly P Branigan. 1998. The representation of verbs: Evidence from syntactic priming in language production. *Journal of Memory and Language*, 39(4):633–651.

- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international joint conference on Artificial intelligence, IJCAI'09*, pages 1199–1204, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Dan Roth and Wen-tau Yih. 2004. *A linear programming formulation for global inference in natural language tasks*. Defense Technical Information Center.
- John Sinclair. 1991. *Corpus, concordance, collocation*. Describing English language. Oxford University Press.
- Anatol Stefanowitsch and Stefan Th Gries. 2003. Collocations: Investigating the interaction of words and constructions. *International journal of corpus linguistics*, 8(2):209–243.
- Philip J. Stone and Earl B. Hunt. 1963. A computer approach to content analysis: studies using the general inquirer system. In *Proceedings of the May 21-23, 1963, spring joint computer conference, AFIPS '63 (Spring)*, pages 241–256, New York, NY, USA. ACM.
- Michael Stubbs. 1995. Collocations and semantic profiles: on the cause of the trouble with quantitative studies. *Functions of language*, 2(1):23–55.
- Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *In EMNLP/VLC 2000*, pages 63–70.
- Peter Turney. 2001. Mining the web for synonyms: Pmi-ir versus lsa on toefl.
- Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation (formerly Computers and the Humanities)*, 39(2/3):164–210.
- Theresa Wilson, Paul Hoffmann, Swapna Somasundaran, Jason Kessler, Janyce Wiebe, Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005a. Opinionfinder: a system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, pages 34–35, Morristown, NJ, USA. Association for Computational Linguistics.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005b. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354, Morristown, NJ, USA. Association for Computational Linguistics.
- Rui Xie and Chunping Li. 2012. Lexicon construction: A topic model approach. In *Systems and Informatics (ICSAI), 2012 International Conference on*, pages 2299–2303. IEEE.
- Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. In *Technical Report CMU-CALD-02-107*. CarnegieMellon University.

# Author Index

- A.R, Balamurali, 412  
Abend, Omri, 228  
Adamson, David, 104  
Aker, Ahmet, 402  
Alfonseca, Enrique, 1243  
Allauzen, Cyril, 43  
Allen, James, 64  
Almeida, Miguel, 196  
Andreas, Jacob, 924  
Andrew Y., Ng, 455  
Angeli, Gabor, 83  
Aransa, Walid, 832  
Arase, Yuki, 1597  
Arka, I Wayan, 550
- Baldrige, Jason, 583, 1466  
Baldwin, Tyler, 1159  
Bali, Kalika, 1713  
Bamman, David, 352  
Barbosa, Denilson, 1312  
Baroni, Marco, 1517  
Bartie, Phil, 1660  
Barzilai, Regina, 291, 1294  
Bauer, Daniel, 924  
Bauer, John, 455  
Beigman Klebanov, Beata, 1148  
Berant, Jonathan, 1331  
Berg-Kirkpatrick, Taylor, 207  
Bergsma, Shane, 710  
Berwick, Robert, 1321  
Bhattacharyya, Pushpak, 412  
Black, Alan, 176  
Blunsom, Phil, 894  
Bohus, Dan, 466  
Bond, Francis, 1352  
Boros, Tiberiu, 692  
Börschinger, Benjamin, 1508  
Braune, Fabienne, 811  
Büchse, Matthias, 145  
Butt, Miriam, 550
- Cai, Qingqing, 423  
Callison-Burch, Chris, 1374  
Cardie, Claire, 1384, 1395, 1640
- Carenini, Giuseppe, 486  
Carpuat, Marine, 1435  
Castelli, Vittorio, 1384  
Celikyilmaz, Asli, 914  
Cer, Daniel, 311  
Cetinoglu, Ozlem, 550  
Chang, Ming-Wei, 1744  
Chao, Lidia S., 770  
Charniak, Eugene, 302  
Che, Wanxiang, 125, 1073  
Chen, Boxing, 1285  
Chen, Wenliang, 434  
Chen, Yubo, 1764  
Cheung, Jackie Chi Kit, 392, 1233  
Chiang, David, 924  
Choi, Jinho D., 1052  
Choi, Yejin, 1774  
Choudhury, Monojit, 1713  
Choudhury, Pallavi, 1669  
Cohen, Shay B., 1033  
Cohn, Trevor, 32, 333, 780, 993  
Creamer, Germán G., 873  
Croft, W. Bruce, 507  
Cuayáhuitl, Heriberto, 1254
- Dagan, Ido, 1331  
Dai, Decheng, 1723  
Dalmas, Tiphaine, 1660  
Danescu-Niculescu-Mizil, Cristian, 250, 1650  
Darwish, Kareem, 1558  
Dasgupta, Anirban, 1014  
Daume III, Hal, 1435  
De Benedictis, Flavio, 528  
De Saeger, Stijn, 1619, 1733  
De Smedt, Koenraad, 550  
Demuth, Katherine, 1508  
Denis, Pascal, 497  
Dethlefs, Nina, 1254  
Di Eugenio, Barbara, 270  
Diab, Mona, 239  
Dickinson, Anna, 1660  
Dione, Cheikh Bamba, 550  
Dolan, Bill, 1669  
Downey, Doug, 343

Durrett, Greg, 114, 207  
Dyer, Chris, 176  
Dyvik, Helge, 550

Eidelman, Vladimir, 1116  
Eisner, Jason, 444  
Endriss, Ulle, 539  
Etzioni, Oren, 1608

Fader, Anthony, 1608  
Faralli, Stefano, 528  
Feng, Minwei, 322  
Feng, Song, 1774  
Feng, Yang, 333  
Fernández, Raquel, 539  
Ferrari, Vittorio, 572  
Ferret, Olivier, 561  
Ferschke, Oliver, 721  
Flati, Tiziano, 1222  
Flor, Michael, 1148  
Florian, Radu, 1384  
Fokkens, Antske, 1691  
Foster, George, 1285  
Foster, Ryan, 1352  
Fournier, Chris, 1702  
Fraser, Alexander, 593  
Freire, Nuno, 1691

Gaizauskas, Rob, 402  
Gao, Yang, 1723  
Garrette, Dan, 583  
Garrido, Guillermo, 1243  
Gildea, Daniel, 64  
Globerson, Amir, 291  
Goldberger, Jacob, 1331  
Gormley, Matthew R., 444  
Goto, Isao, 155  
Green, Spence, 311  
Guinaudeau, Camille, 93  
Guo, Weiwei, 239  
Gurevych, Iryna, 721, 1363

Haffari, Gholamreza, 412, 780  
Haffari, Reza, 1105  
Hagen, Matthias, 1212  
Hakkani-Tur, Dilek, 914  
Hall, David, 114  
Han, Jiawei, 1083  
Hanaoka, Hiroki, 1042  
Hartmann, Silvana, 1363  
Hasegawa, Takayuki, 964  
Hashimoto, Chikara, 1619, 1733  
Hassan, Hany, 1577

Hastie, Helen, 1254  
He, Hua, 1312  
He, Shizhu, 852  
Henry, Katharine, 1435  
Hermann, Karl Moritz, 894, 924  
Ho, Howard, 1159  
Hopkins, Mark, 1416  
Hovy, Eduard, 372  
Howald, Blake, 1406  
Huang, Hongzhao, 1083  
Huang, Liang, 73  
Hwang, Seung-won, 631

Idiart, Marco, 1321  
Inui, Kentaro, 382  
Ion, Radu, 692  
Irvine, Ann, 1435

Jagarlamudi, Jagadeesh, 1435  
Janarthanam, Srinivasan, 1660  
Ji, Heng, 73, 239, 604, 1083  
Jiang, Wenbin, 761, 1063  
Johnson, Mark, 1033, 1508  
Jones, Bevan, 924  
Joty, Shafiq, 486  
Jurafsky, Dan, 250, 1650  
Jurgens, David, 1341

Kaji, Nobuhiro, 964  
Kan, Min-Yen, 731  
Kang, Jun Seok, 1774  
Kauchak, David, 1537  
Kawai, Takao, 1619  
Khapra, Mitesh M., 1275  
Kim, Joohyun, 218  
Kim, Young-Bum, 1527  
Kimelfeld, Benny, 1159  
King, Tracy Holloway, 550  
Kit, Chunyu, 622  
Klein, Dan, 114, 207  
Knight, Kevin, 924  
Koehn, Philipp, 1374  
Koller, Alexander, 145  
Kondadadi, Ravi, 1406  
Kondrak, Grzegorz, 1312  
Korhonen, Anna, 862  
Kozareva, Zornitsa, 682  
Kozhevnikov, Mikhail, 1190  
Kuhn, Jonas, 1547  
Kuhn, Roland, 1285  
Kumar, Ravi, 1014  
Kundu, Gourab, 905

Kurohashi, Sadao, 155  
Kuznetsova, Polina, 1774

Laczkó, Tibor, 550  
Lai, Siwei, 1764  
Lamos, Vasileios, 993  
Lan, Man, 476, 983  
Laparra, Egoitz, 1180  
Lapata, Mirella, 572  
Lassalle, Emmanuel, 497  
Lazaridou, Angeliki, 1517, 1630  
Lee, Taesung, 631  
Lei, Tao, 1294  
Lemon, Oliver, 1254, 1660  
Leskovec, Jure, 250  
Li, Chen, 1004  
Li, Fangtao, 1723  
Li, Guofu, 660  
Li, Haibo, 604  
Li, Hao, 239  
Li, He, 1083  
Li, Juanzi, 641  
Li, Mu, 166, 752  
Li, Peifeng, 1477  
Li, Qi, 73, 604  
Li, Yitong, 1304  
Li, Yunyao, 1159  
Li, Zhixing, 641  
Ling, Wang, 176  
Liu, Bing, 671, 1680  
Liu, Fang, 852  
Liu, Kai, 1063  
Liu, Kang, 1754, 1764  
liu, lemao, 791  
Liu, Qun, 761, 1063  
Liu, Shujie, 166  
Liu, Ting, 125  
Liu, Xiaohua, 1304  
Liu, Xingkun, 1660  
Liu, Yang, 1, 852, 1004  
Liu, Ziqi, 1169  
Long, Fan, 1294  
Lopez, Adam, 1374  
Lü, Yajuan, 761, 1063  
Lu, Yi, 1304  
Lucas, Michael, 343  
Lugaresi, Camillo, 270  
Luo, Xiaoqiang, 822

Mackaness, William, 1660  
Maletti, Andreas, 811  
Malioutov, Igor, 1321

Manning, Christopher D., 311, 455, 1073  
Manshadi, Mehdi, 64  
Mareček, David, 281, 517  
Marelli, Marco, 1517  
Martins, Andre, 196  
Marton, Yuval, 1116  
Matsuzaki, Takuya, 1042  
Maxwell, K. Tamsin, 507  
May, Jonathan, 1416  
Mayfield, Elijah, 104  
McCallum, Andrew, 1052  
Meek, Christopher, 1744  
Mehdad, Yashar, 486  
Melamud, Oren, 1331  
Menezes, Arul, 1577  
Meraz, Sharon, 1680  
Metallinou, Angeliki, 466  
Meurer, Paul, 550  
Mielens, Jason, 583  
Mihalcea, Rada, 973  
Mima, Hideki, 1042  
Mistica, Meladel, 550  
Miyao, Yusuke, 1042  
Mizuno, Junta, 382  
Mohtarami, Mitra, 983  
Mooney, Raymond, 218  
Morency, Louis-Philippe, 973  
Morita, Hajime, 1023  
Moschitti, Alessandro, 1498  
Mukherjee, Arjun, 671, 1680  
Munaka, Tatsuji, 884

Nagata, Ryo, 1137  
Nakashole, Ndapandula, 1488  
Narisawa, Katsuma, 382  
Nastase, Vivi, 651  
Navigli, Roberto, 528, 1222, 1341  
Ney, Hermann, 322, 615, 1568  
Ng, Hwee Tou, 1456  
Ng, Raymond, 486  
Ng, Vincent, 260  
Nguyen, ThuyLinh, 1587  
Niu, Zhengyu, 476  
Nivre, Joakim, 135  
Nuhn, Malte, 615, 1568

O'Connor, Brendan, 352, 1094  
Oberlander, Jon, 507  
Oh, Jong-Hoon, 1619, 1733  
Ohtake, Kiyonori, 1619, 1733  
Okazaki, Naoaki, 382  
Okumura, Manabu, 841, 1023

Özbal, Gözde, 1446

Padó, Sebastian, 1201  
Paramita, Monica, 402  
Passeigneur, Rebecca J., 873  
Pastusiak, Andrzej, 1744  
Patejuk, Agnieszka, 550  
Pedersen, Ted, 1691  
Penn, Gerald, 392, 1233  
Penstein Rosé, Carolyn, 104  
Perez-Rosas, Veronica, 973  
Persing, Isaac, 260  
Peter, Jan-Thorsten, 322  
Pighin, Daniele, 1243, 1446  
Pilehvar, Mohammad Taher, 1341  
Plamada, Magdalena, 1374  
Plank, Barbara, 1498  
Poon, Hoifung, 933  
Popat, Kashyap, 412  
Popel, Martin, 517  
Postma, Marten, 1691  
Potthast, Martin, 1212  
Potts, Christopher, 250  
Preoțiu-Pietro, Daniel, 993

Qian, Xian, 1004  
Quan, Xiaojun, 622  
Quernheim, Daniel, 811  
Quirk, Chris, 1669

Radziszewski, Adam, 701  
Raghavan, Hema, 1384  
Rákosi, György, 550  
Ramanath, Rohan, 1713  
Ramanathan, Ananthakrishnan, 1275  
Rappoport, Ari, 228  
Ravi, Sujith, 362, 1014  
Razmara, Majid, 1105  
Recasens, Marta, 1650  
Reichart, Roi, 862  
Resnik, Philip, 1116  
Rigau, German, 1180  
Riley, Michael, 43  
Rinard, Martin, 1294  
Rittberger, Marc, 721  
Roark, Brian, 43  
Rokhlenko, Oleg, 742  
Rosén, Victoria, 550  
Roth, Dan, 905  
Rudinger, Rachel, 1435  
Rudzicz, Frank, 944

Sagisaka, Yoshinori, 884

Saha Roy, Rishiraj, 1713  
Saint-Amand, Herve, 1374  
Sano, Motoki, 1619, 1733  
Sarikaya, Ruhi, 914  
Sarkar, Anoop, 1105  
Sartorio, Francesco, 135  
Sasano, Ryohei, 1023  
Satta, Giorgio, 135  
Schamper, Julian, 1568  
Scheible, Christian, 954  
Schilder, Frank, 1406  
Schoenemann, Thomas, 22  
Schulte im Walde, Sabine, 593  
Schütze, Hinrich, 954  
Schwenk, Holger, 832  
Seemann, Nina, 811  
Sennrich, Rico, 832  
Setiawan, Hendra, 1264  
Shen, Libin, 1264  
Shiba, Mitsuteru, 884  
Shieber, Stuart, 302  
Si, Xiance, 1723  
Siahbani, Maryam, 1105  
Silberer, Carina, 572  
Siskind, Jeffrey Mark, 53  
Smith, Jason R., 1374  
Smith, Noah A., 352, 1094  
Šnajder, Jan, 1201  
Snyder, Benjamin, 1527  
Socher, Richard, 455  
Song, Yan, 622  
Specia, Lucia, 32  
Speriosu, Michael, 1466  
Sporleder, Caroline, 1630  
Srikumar, Vivek, 905  
Štěpánek, Jan, 517  
Stein, Benno, 1212  
Stewart, Brandon M., 1094  
Straka, Milan, 281  
Strapparava, Carlo, 651, 1446  
Strube, Michael, 93  
Su, Keh-Yih, 11  
Sudhof, Moritz, 250  
Sulger, Sebastian, 550  
Sumita, Eiichiro, 155, 791, 802, 841  
Sun, Meng, 761  
Sun, Yizhou, 1083  
Swanson, Ben, 302  
Szpektor, Idan, 742, 1331

Takamura, Hiroya, 841, 1023  
Tamura, Akihiro, 155, 841

Tan, Chew Lim, 983  
Tang, Jie, 641  
Tanigaki, Koichi, 884  
Tian, Zhenhua, 1169  
Titov, Ivan, 1190, 1630  
Torisawa, Kentaro, 1619, 1733  
Toyoda, Masashi, 964  
Trancoso, Isabel, 176, 770  
Tratz, Stephen, 372  
Tufis, Dan, 692  
Tur, Gokhan, 914  
Tylenda, Tomasz, 1488  
  
Uematsu, Sumire, 1042  
Uszkoreit, Jakob, 83  
Utiyama, Masao, 155  
  
Van Durme, Benjamin, 710  
van Erp, Marieke, 1691  
Varga, István, 1619  
Veale, Tony, 660  
Venkataraman, Vivek, 1680  
Villavicencio, Aline, 1321  
Visweswariah, Karthik, 1275  
Vogel, Stephan, 1587  
Vogler, Heiko, 145  
Volkova, Svitlana, 1669  
Völske, Michael, 1212  
Vossen, Piek, 1691  
  
Wang, Aobo, 731  
Wang, Kun, 11  
Wang, Lu, 1384, 1395  
Wang, Mengqiu, 1073  
Wang, Sida, 311  
Wang, Wen, 604  
Wang, Zhigang, 641  
Watanabe, Taro, 791, 802, 841  
Watanabe, Yotaro, 382  
Webber, Bonnie, 1660  
Wei, Furu, 1304  
Weikum, Gerhard, 1488  
Weller, Marion, 593  
Wen, Zhen, 1083  
Whittaker, Edward, 1137  
Williams, Jason, 466  
Wong, Derek F., 770  
Wu, Haocheng, 1304  
Wu, Leon, 873  
Wu, Shuangzhi, 752  
Wu, Yuanbin, 1456  
  
Xiang, Bing, 822, 1264  
  
Xiang, Guang, 176  
Xiang, Hengheng, 1169  
Xie, Boyi, 873  
Xu, Liheng, 1754, 1764  
Xu, Yu, 476  
  
Yamangil, Elif, 302  
Yancheva, Maria, 944  
Yang, Bishan, 1640  
Yang, Nan, 166, 752  
Yang, Yating, 761  
Yates, Alexander, 423  
Yih, Wen-tau, 1744  
Yoshinaga, Naoki, 964  
Yu, Dian, 1083  
Yu, Haonan, 53  
Yu, Nenghai, 166  
  
Z. Pan, Jeff, 641  
Žabokrtský, Zdeněk, 517  
Zamparelli, Roberto, 1517  
Zarrieß, Sina, 1547  
Zeller, Britta, 1201  
Zeman, Daniel, 517  
Zeng, Xiaodong, 770  
Zettlemoyer, Luke, 1608, 1669  
Zhai, Feifei, 1127  
Zhang, Bo, 187  
Zhang, Congle, 1159  
Zhang, Dongdong, 752  
Zhang, Jiajun, 1127, 1425  
Zhang, Meishan, 125  
Zhang, Min, 434  
Zhang, Yuan, 291  
Zhang, Yue, 125, 434  
Zhao, Jun, 852, 1754, 1764  
Zhao, Tiejun, 791, 802  
Zheng, Jing, 604  
Zheng, Qinghua, 1169  
Zheng, Xun, 187  
Zhou, Bowen, 822, 1264  
Zhou, Guangyou, 852  
Zhou, Guodong, 1477  
Zhou, Ming, 166, 1304, 1597  
Zhou, Shuchang, 1723  
Zhou, Yu, 1127  
Zhu, Conghui, 802  
Zhu, Jingbo, 434  
Zhu, Jun, 187  
Zhu, Muhua, 434  
Zhu, Qiaoming, 1477  
Zong, Chengqing, 11, 1127, 1425